



**ITS**  
Institut  
Teknologi  
Sepuluh Nopember

**TUGAS AKHIR - TE 141599**

**ANALISA METODE PENGATURAN ADAPTIF UNTUK  
PENALAAN OTOMATIS PARAMETER KONTROLER PID  
PADA SISTEM *SUPERVISORY CONTROL AND DATA  
ACQUISITION* (SCADA)**

Adetya Devritama  
NRP 2212100015

Dosen Pembimbing  
Ir. Ali Fatoni, M.T.  
Imam Arifin, S.T., M.T.

JURUSAN TEKNIK ELEKTRO  
Fakultas Teknologi Industri  
Institut Teknologi Sepuluh Nopember  
Surabaya 2016



**FINAL PROJECT - TE 141599**

**ADAPTIVE CONTROL METHOD ANALYSIS FOR AUTO  
TUNING PID CONTROLLER PARAMETER IN  
SUPERVISORY CONTROL AND DATA ACQUISITION  
(SCADA) SYSTEM**

Adetya Devritama  
NRP 2212100015

Supervisor  
Ir. Ali Fatoni, M.T.  
Imam Arifin, S.T., M.T.

ELECTRICAL ENGINEERING DEPARTMENT  
Faculty of Industrial Technology  
Sepuluh Nopember Institute of Technology  
Surabaya 2016

**ANALISA METODE PENGATURAN ADAPTIF UNTUK  
PENALAAN OTOMATIS PARAMETER KONTROLER PID  
PADA SISTEM SUPERVISORY CONTROL AND DATA  
ACQUISITION (SCADA)**

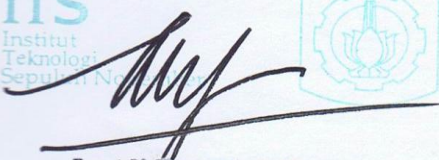
**TUGAS AKHIR**

**Diajukan Guna Memenuhi Sebagian Persyaratan  
Untuk Memperoleh Gelar Sarjana Teknik  
Pada**

**Bidang Studi Teknik Sistem Pengaturan  
Jurusan Teknik Elektro  
Institut Teknologi Sepuluh Nopember**

**Menyetujui :**

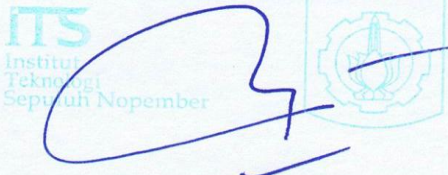
**Dosen Pembimbing I**



**Ir. Ali Fatoni, MT.**

**NIP. 196206031989031002**

**Dosen Pembimbing II**



**Imam Arifin, ST., MT.**

**NIP. 197302222002121001**



# **Analisa Metode Pengaturan Adaptif Untuk Penalaan Otomatis Parameter Kontroler PID Pada Sistem *Supervisory Control and Data Acquisition* (SCADA)**

**Penulis** : Adetya Devritama  
**Pembimbing I** : Ir. Ali Fatoni, M.T.  
**Pembimbing II** : Imam Arifin, S.T., M.T.

## **ABSTRAK**

Pengetahuan terhadap sistem SCADA merupakan hal yang penting untuk dikuasai bagi calon insinyur. Pada kenyataanya, fasilitas pengenalan sistem SCADA bagi para calon insinyur masih sangat kurang. Banyak pelaku industri yang menerapkan sistem SCADA untuk berbagai pengaturan proses yang mempunyai dinamika beban, salah satunya adalah pengaturan level. Dengan adanya dinamika pada sistem tersebut, karakteristik dari sistem juga akan berubah, sehingga sinyal kontrol harus disesuaikan dengan kebutuhan sistem untuk mendapatkan respon yang diinginkan. Pada Tugas Akhir ini akan dirancang miniatur sistem SCADA sebagai fasilitas pembelajaran bagi para calon insinyur. Pada sistem yang dibuat diterapkan metode PID dengan algoritma adaptasi untuk mengatur sistem pengaturan level. Metode ini dapat melakukan penalaan parameter kontroler secara otomatis sesuai dengan keadaan sistem pada saat terjadi pembebanan, sehingga mampu meminimalkan kesalahan waktu tunak lebih baik jika dibandingkan kontroler PID penalaan Cohen Coon dengan RMSE sebesar 5,05%, dan mempunyai waktu kembali lebih cepat, yaitu 18 detik untuk beban nominal, dan 23 detik untuk beban maksimal.

**Kata kunci:** Kontroler adaptif, kontroler PID, pengaturan level, Sistem SCADA, penalaan otomatis

# ***Adaptive Control Method Analysis for Auto Tuning PID Controller Parameter in Supervisory Control and Data Acquisition (SCADA) System***

**Author** : Adetya Devritama  
**Supervisor I** : Ir. Ali Fatoni, M.T.  
**Supervisor II** : Imam Arifin, S.T., M.T.

## **ABSTRACT**

*The knowledge on SCADA system is essential for engineer candidate. In fact, facilities to learn SCADA system for engineer candidates is still limited. Many industry subjects applied SCADA system to control processes with dynamic load, in example water level control. The characteristic of the system will change due to change of the load, so that the control signal must available with system's requirements to obtain desired respond. Nowadays, there are many methods proposed to solve this problem, include auto tuning PID controller. This final project is aimed to build SCADA system model to improve knowledge of SCADA system for engineer candidates. In this SCADA system, PID control method based on adaptation algorithm is applied to control water level. This method is based on mathematical analysis with output control signal which posses adaptation algorithm so that the parameter value can be adjusted automatically. This method can be used for auto tuning PI controller parameter when the system's load changes, so it is more effective to minimize steady state error (5,03%) and give faster recovery time (18 seconds at nominal load, and 23 seconds at maximal load).*

**Key words:** *Adaptive controller, PID controller, level control, SCADA system, auto tuning*

## DAFTAR ISI

HALAMAN JUDUL.....	i
PERNYATAAN KEASLIAN TUGAS AKHIR.....	iii
HALAMAN PENGESAHAN.....	v
ABSTRAK.....	vii
ABSTRACT.....	ix
KATA PENGANTAR .....	xi
DAFTAR ISI.....	xiii
DAFTAR GAMBAR .....	xvii
DAFTAR TABEL.....	xix
 BAB 1    PENDAHULUAN .....	 1
1.1    Latar Belakang .....	1
1.2    Rumusan Masalah .....	2
1.3    Batasan Masalah .....	2
1.4    Tujuan .....	2
1.5    Metodologi.....	3
1.6    Sistematika.....	3
1.7    Relevansi.....	4
 BAB 2    PENGATURAN LEVEL TANGKI PADA SISTEM SCADA .....	 5
2.1    Sistem <i>Supervisory Control and Data Acquisition</i> (SCADA) .....	5
2.2    Sistem Pengaturan Melalui Jaringan .....	6
2.3    Pengenalan <i>Plant</i> PCT 100 [5].....	7
2.3.1    Process rig pada PCT 100 .....	8
2.3.2    Control module pada PCT 100.....	9
2.4    Kontroler PID.....	10
2.5    Metode Penalaan Parameter PID.....	10
2.5.1    Aturan Penalaan Ziegler-Nichols .....	10
2.5.2    Aturan Penalaan Cohen Coon .....	12
2.6    Kontroler Adaptif.....	14
2.7 <i>Model Reference Adaptive System</i> (MRAS) .....	15
2.8    Pemodelan Matematis Sistem Pengaturan Level .....	16
2.8.1    Pemodelan Gain Pompa .....	17
2.8.2    Pemodelan Sistem Pengaturan Level Pada Tangki .....	17
2.8.3    Pemodelan Gain pada Sensor.....	18

BAB 3	PERANCANGAN SISTEM.....	21
3.1	Perancangan Sistem.....	21
3.1.1	Perancangan Arsitektur Sistem.....	21
3.1.2	Konfigurasi Komunikasi pada Sistem SCADA.....	22
3.2	Perancangan Human Machine Interface (HMI).....	22
3.2.1	Perancangan HMI <i>Home Window</i> .....	22
3.2.2	Perancangan HMI <i>Local test Window</i> .....	23
3.2.3	Perancangan HMI <i>Controller setting Window</i> .....	24
3.2.4	Perancangan HMI <i>Alarm Window</i> .....	24
3.3	Perancangan Kontroler .....	25
3.3.1	Perancangan Kontroler PI dengan Metode Cohen Coon .....	25
3.3.2	Perancangan Penalaan otomatis PI dengan <i>Model Reference Adaptive Control</i> (MRAC) .....	28
BAB 4	PENGUJIAN DAN ANALISIS .....	35
4.1	Simulasi Pengujian Variasi Keadaan pada Penalaan PI Cohen Coon .....	35
4.1.1	Pengujian Variasi <i>Set point</i> .....	35
4.1.2	Pengujian Variasi <i>Kp</i> .....	36
4.1.3	Pengujian Variasi <i>Ti</i> .....	37
4.1.4	Pengujian Variasi Derau .....	37
4.1.5	Pengujian Variasi Beban .....	39
4.2	Simulasi Pengujian Variasi Keadaan pada Metode Penalaan otomatis PI dengan MRAC.....	40
4.2.1	Pengujian Variasi <i>Set point</i> .....	40
4.2.2	Pengujian Variasi Konstanta Adaptasi .....	41
4.2.3	Pengujian Variasi Derau .....	42
4.2.4	Pengujian Variasi Beban .....	43
4.2.5	Perhitungan RMSE pada Metode Kontrol PI dengan Penalaan Cohen Coon .....	44
4.2.6	Perhitungan RMSE pada Metode Penalaan otomatis dengan MRAC .....	44
4.3	Implementasi Kontroler pada Sistem Pengaturan Level .....	46
4.3.1	Pengujian Variasi Konstanta Adaptasi .....	46
4.3.2	Pengujian Variasi <i>Set point</i> .....	47
4.3.3	Pengujian Variasi Beban .....	49
4.3.4	Adaptasi Nilai <i>Kp</i> dan <i>Ki</i> .....	50

BAB 5	PENUTUP.....	53
5.1	Kesimpulan .....	53
5.2	Saran .....	53
DAFTAR PUSTAKA .....		55
LAMPIRAN.....		57
BIODATA PENULIS .....		77



## DAFTAR GAMBAR

Gambar 2.1	Struktur arsitektur dari sistem SCADA .....	5
Gambar 2.2	Skema komunikasi data pada sistem pengaturan berjaringan .....	7
Gambar 2.3	Tampilan fisik process rig pada PCT 100[5].....	8
Gambar 2.4	Diagram blok sistem dengan penambahan kontroler PID .....	10
Gambar 2.5	Grafik respon kurva S pada hasil pengujian sistem.....	11
Gambar 2.6	Osilasi yang terjadi ketika nilai gain kritis .....	12
Gambar 2.7.	Pengujian step untuk mendapatkan kurva S sebagai dasar penalaan Cohen Coon .....	13
Gambar 2.8	Diagram blok metode sistem pengaturan adaptif .....	14
Gambar 2.9	Diagram blok dari MRAS .....	15
Gambar 2.10.	Skema proses pengaturan level pada tangki .....	16
Gambar 2.11.	Ilustrasi sistem pengaturan level pada tangki .....	16
Gambar 2.12	Respon sistem pengaturan level loop terbuka .....	19
Gambar 3.1.	Arsitektur sistem SCADA untuk pengaturan level dan kecepatan motor DC.....	21
Gambar 3.12.	Tampilan antarmuka pada HMI home window .....	23
Gambar 3.13.	Tampilan antarmuka pada HMI local test window .....	23
Gambar 3.14.	Tampilan HMI controller setting window .....	24
Gambar 3.15.	Tampilan antarmukan pada HMI alarm window.....	25
Gambar 3.16	Diagram blok sistem dengan kontroler PI .....	25
Gambar 3.17	Hasil respon sistem dengan kontroler PI penalaan Cohen Coon .....	27
Gambar 3.18	Diagram sistem metode penalaan otomatis PI dengan MRAC .....	28
Gambar 3.19	Diagram blok untuk algoritma penalaan otomatis parameter PI .....	32
Gambar 3.20	Respon pengujian metode penalaan otomatis PI dengan MRAC .....	32
Gambar 4.1	Hasil pengujian variasi <i>set point</i> pada metode control PI hasil penalaan dengan penalaan otomatis.....	35
Gambar 4.2	Hasil pengujian variasi nilai $K_p$ pada metode PI penalaan Cohen Coon .....	36

Gambar 4.3	Pengujian variasi $T_i$ pada metode PI penalaan Cohen Coon .....	37
Gambar 4.4	Diagram blok untuk penambahan derau dan disturbance pada sistem .....	38
Gambar 4.5	Hasil pengujian dengan penambahan variasi derau pada sistem .....	38
Gambar 4.6	Hasil penambahan variasi disturbance pada sistem .....	39
Gambar 4.7	Hasil pengujian variasi <i>set point</i> pada metode penalaan otomatis dengan MRAC .....	40
Gambar 4.8	Hasil pengujian variasi konstanta adaptasi .....	41
Gambar 4.9	Hasil pengujian penambahan variasi derau pada sistem dengan metode control penalaan otomatis PI menggunakan MRAC .....	42
Gambar 4.10	Hasil pengujian variasi beban pada metode penalaan otomatis PI .....	43
Gambar 4.11.	Set up komponen untuk pengambilan data implementasi sistem pengaturan level .....	46
Gambar 4.12.	Pengujian variasi konstanta adaptasi pada sistem pengaturan level .....	47
Gambar 4.13.	Pengujian <i>set point</i> tracking pada sistem pengaturan level .....	48
Gambar 4.14.	Pengujian <i>set point</i> tracking pada sistem pengaturan level .....	49
Gambar 4.15.	(a) Adaptasi nilai $K_p$ dan (b) adaptasi nilai $K_i$ pada metode kontrol penalaan otomatis PI dengan MRAC ..	50

## DAFTAR TABEL

Tabel 2.1	Penjelasan bagian-bagian pada process rig PCT 100[5].....	9
Tabel 2.2	Penjelasan fungsi fault switch pada control module PCT 100[5] .....	9
Tabel 2.3	Tabel aturan penalaan metode pertama Ziegler Nichols .....	11
Tabel 2.4	Tabel aturan penalaan metode kedua Ziegler Nichols.....	12
Tabel 2.5	Tabel aturan penalaan metode Cohen Coon .....	14
Tabel 2.6	Parameter-parameter pada sistem.....	17
Tabel 3.1.	Nilai ketinggian level tangki pada setiap message list .....	24
Tabel 3.2	Parameter-parameter pada respon sistem open loop .....	26
Tabel 4.1.	Nilai RMSE pada sistem dengan metode kontrol PI penalaan Cohen Coon untuk setiap variasi kondisi .....	44
Tabel 4.2.	Perhitungan nilai RMSE untuk setiap variasi kondisi .....	45
Tabel 4.3.	Perhitungan <i>rise time</i> dan RMSE untuk setiap nilai konstanta adaptasi.....	47
Tabel 4.4.	Perhitungan nilai RMSE untuk variasi set point.....	48
Tabel 4.5.	Perhitungan waktu kembali untuk setiap kondisi beban.....	50

## BAB 3

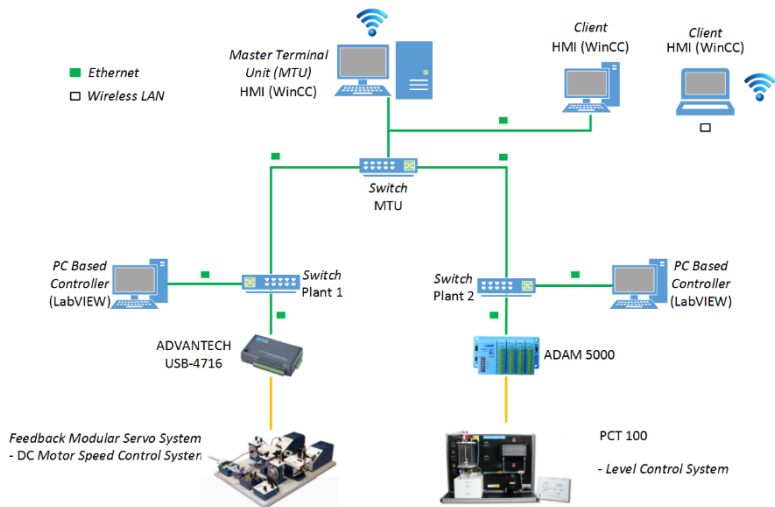
### PERANCANGAN SISTEM

#### 3.1 Perancangan Sistem

Terdapat beberapa tahapan yang dilakukan pada perancangan sistem SCADA. Tahapan perancangan sistem yang dilakukan antara lain perancangan arsitektur sistem, perancangan Human Machine Interface (HMI), perancangan kontroler.

##### 3.1.1 Perancangan Arsitektur Sistem

Arsitektur untuk pengaturan level pada Tugas Akhir ini akan dirancang dengan metode *supervisory control*, di mana proses pengendalian dan pengawasan dari sistem dilakukan pada level *supervisory*. Arsitektur dari sistem yang akan dirancang ditampilkan pada Gambar 3.1



**Gambar 3.1.** Arsitektur sistem SCADA untuk pengaturan level dan kecepatan motor DC

Pada arsitektur SCADA yang dirancang terdapat dua *plant* yang diatur pada level *field device*, yaitu PCT-100 sebagai model level

pengaturan level dan *modular servo feedback system* untuk model pengaturan kecepatan motor DC. Data-data dari *field device* tersebut diakuisisi oleh modul ADAM 5000 untuk selanjutnya diteruskan menuju *local controller*.

Pada level *local controller*, terdapat program yang berfungsi mengatur jalannya proses pada *field device*. Data-data dari *local controller* diteruskan menuju kontroler pusat pada level *supervisory* melalui komunikasi Ethernet. Pada kontroler pusat terdapat *Human Machine Interface* (HMI) yang berfungsi sebagai antarmuka antara mesin dengan proses yang sedang berlangsung. Data-data pada *server* akan diakses oleh komputer *client* yang dilakukan oleh operator maupun pemilik *plant* untuk melakukan pengawasan maupun pengendalian terhadap kondisi *plant* di lapangan.

### **3.1.2 Konfigurasi Komunikasi pada Sistem SCADA**

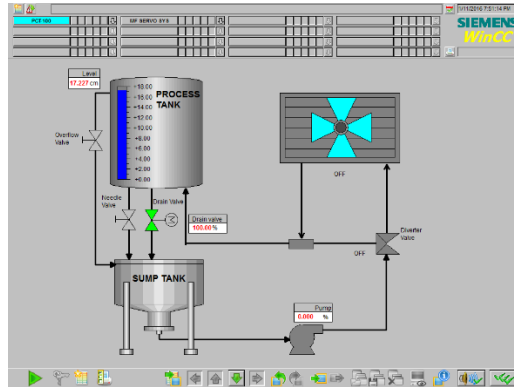
Pada rancangan sistem SCADA yang dibangun, terdapat beberapa konfigurasi untuk menjalankan keseluruhan sistem, mulai dari konfigurasi *field device* ke *local controller*, hingga konfigurasi *local controller* ke *supervisory controller*. Tahapan konfigurasi sistem SCADA yang dilakukan secara detail ditampilkan pada Lampiran.

## **3.2 Perancangan Human Machine Interface (HMI)**

Secara umum, HMI merupakan antarmuka antara operator dengan elemen-elemen dari proses pengaturan yang sedang berlangsung. Pada sistem SCADA yang dirancang, terdapat beberapa tampilan HMI yang digunakan dengan berbagai fungsi masing-masing, mulai dari fungsi pengawasan, pengendalian, *reporting*, dan *alarm message*. Tampilan HMI yang dirancang pada sistem SCADA mencakup fungsi-fungsi tersebut terdiri dari HMI *Home Window*, *Local test Window*, *Controller setting Window*, dan *Alarm Window*.

### **3.2.1 Perancangan HMI Home Window**

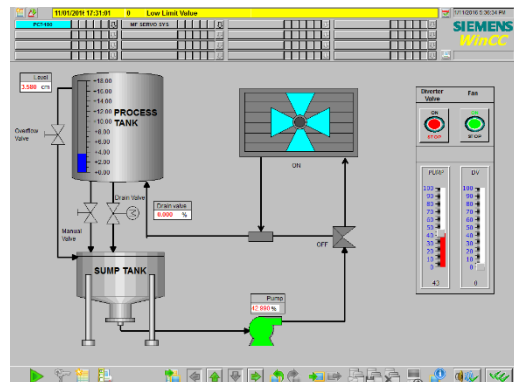
HMI *home window*, ditunjukkan pada Gambar 3.2, berfungsi untuk melakukan proses pengawasan dari keseluruhan kondisi proses yang sedang berjalan. Pada HMI ini terdapat beberapa indikator berupa tulisan dan nyala lampu yang menunjukkan status dari setiap objek yang terdapat pada HMI *home window* seperti pompa, *diverter valve*, *cooling fan*, sensor level, dan *drain valve*.



**Gambar 3.2.** Tampilan antarmuka pada HMI *home window*

### 3.2.2 Perancangan HMI *Local test Window*

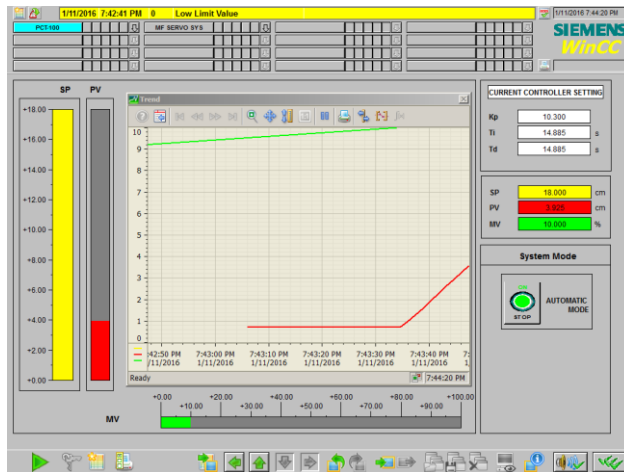
HMI *local test window*, ditunjukkan pada Gambar 3.3, merupakan tampilan antarmuka yang berfungsi sebagai fasilitas untuk operator melakukan pengendalian terhadap proses yang sedang berlangsung secara manual. Tampilan dari HMI *local test* secara umum terdiri dari objek-objek yang sama dengan tampilan HMI *home window* dengan penampahan fasilitas-fasilitas untuk melakukan pengendalian terhadap nilai parameter pada proses yang sedang berlangsung.



**Gambar 3.3.** Tampilan antarmuka pada HMI *local test window*

### 3.2.3 Perancangan HMI Controller setting Window

HMI *controller setting window*, ditunjukkan pada Gambar 3.4., merupakan tampilan antarmuka yang berfungsi sebagai fasilitas bagi operator untuk memberikan nilai-nilai parameter kontroler dan *set point* terhadap proses yang sedang berlangsung. Pada HMI ini juga terdapat trend dan bar yang menampilkan nilai dari parameter sistem.



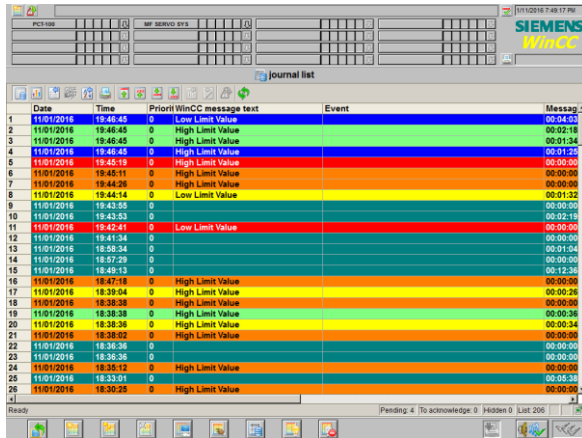
Gambar 3.4. Tampilan HMI *controller setting window*

### 3.2.4 Perancangan HMI Alarm Window

HMI *alarm window*, ditampilkan pada Gambar 3.5, merupakan antarmuka yang mempunyai fasilitas sebagai message log dan report log kepada operator. Dengan adanya *alarm window*, operator dapat mengetahui status dari kondisi proses yang sedang berlangsung. Kondisi status dari sistem diatur berdasar message list seperti pada Tabel 3.1.

Tabel 3.1. Nilai ketinggian level tangki pada setiap message list

No.	Teks Peringatan	Nilai (level)
1	<i>Alarm High</i>	17 cm
2	<i>Warning High</i>	15 cm
3	<i>Warning Low</i>	5 cm
4	<i>Alarm Low</i>	2 cm



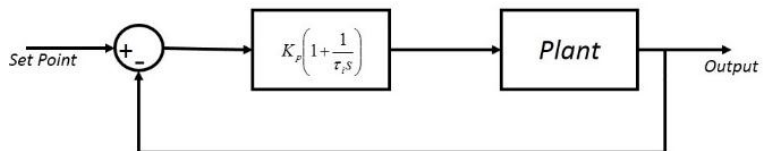
**Gambar 3.5. Tampilan antarmukan pada HMI *alarm window***

### 3.3 Perancangan Kontroler

Berdasarkan hasil pemodelan sistem, diperoleh sebuah sistem dengan kesalahan waktu tunak sebesar 75%, oleh karena itu diperlukan adanya kontroler untuk memperbaiki respon sistem. Respon yang diinginkan adalah respon sistem dengan kesalahan waktu tunak minimal dan tidak ada pengaruh pembebanan.

### 3.3.1 Perancangan Kontroler PI dengan Metode Cohen Coon

Dengan menggunakan metode Cohen Coon, akan dirancang kontroler PI yang dengan tujuan memperoleh respon sistem seperti kriteria yang diinginkan. Maka jika digunakan kontroler PI, diagram blok sistem akan berubah menjadi seperti pada Gambar 3.6.



**Gambar 3.6 Diagram blok sistem dengan kontroler PI**



Berdasarkan Gambar 2.12, didapatkan parameter-parameter yang diperlukan untuk melakukan penalaan parameter  $K_p$  dan  $\tau_i$  dengan menggunakan metode Cohen Coon. Parameter-parameter tersebut ditampilkan pada Tabel 3.2.

**Tabel 3.2 Parameter-parameter pada respon sistem *open loop***

No.	Parameter	Nilai
1	$K$	1.75
2	$T_0$	0 detik
3	$T_2$	81.39 detik
4	$T_3$	115.2 detik

Dari Tabel 3.2, dapat dihitung parameter-parameter untuk menentukan nilai dari  $K_p$  dan  $\tau_i$ , yaitu  $T_1$ ,  $\tau$ ,  $\tau_{del}$ , dan  $r$ . Perhitungan dari masing-masing nilai parameter tersebut dapat dihitung pada Persamaan 3.1 hingga Persamaan 3.4.

$$T_1 = \frac{T_2 - \ln(2) \cdot T_3}{1 - \ln(2)} = 4.97 \quad (3.1)$$

$T_1$  merupakan waktu yang mendekati waktu tunda dari sistem. Parameter  $T_1$  berfungsi untuk menentukan nilai parameter lainnya.

$$\tau = T_3 - T_1 = 110.23 \quad (3.2)$$

di mana  $\tau$  merupakan waktu selisih antara  $T_3$  dan  $T_1$ .

$$\tau_{del} = T_1 - T_0 = 4.97 \quad (3.3)$$

di mana  $\tau_{del}$  merupakan selisih antara  $T_1$  dan  $T_0$ .

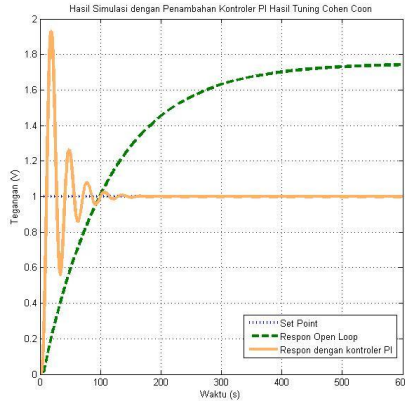
$$r = \frac{\tau_{del}}{\tau} = 0.05 \quad (3.4)$$

di mana  $r$  merupakan perbandingan antara  $\tau_{del}$  dan  $\tau$ .

Setelah mengetahui nilai-nilai dari setiap parameter tersebut, penentuan penalaan untuk nilai  $K_p$  dan  $\tau_i$  berdasarkan aturan penalaan Cohen Coon seperti pada Tabel 2.5. Berdasarkan pada Tabel 2.5, penentuan nilai untuk  $K_p$  dan  $\tau_i$  dari sistem dapat dihitung pada Persamaan 3.5 dan 3.6.

$$K_p = \frac{1}{r \cdot K} \left( 0.9 + \frac{r}{12} \right) = 10.33 \quad (3.5)$$

$$\tau_i = \tau_{del} \cdot \frac{30 + 3 \cdot r}{9 + 20 \cdot r} = 14.98 \text{ detik} \quad (3.6)$$



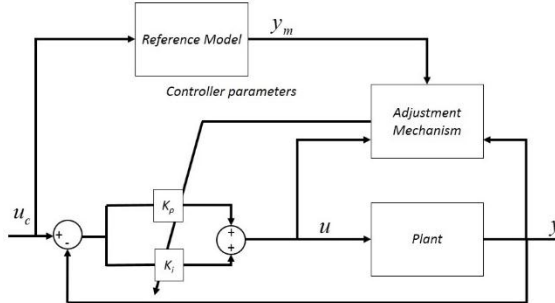
**Gambar 3.7 Hasil respon sistem dengan kontroler PI penalaan Cohen Coon**

Nilai-nilai parameter  $K_p$  dan  $\tau_i$  dari hasil penalaan menggunakan metode Cohen Coon tersebut kemudian dimasukkan pada sistem, sehingga jika disimulasikan akan muncul respon *open loop* sistem dengan kontroler PI seperti ditampilkan pada Gambar 3.7.

$$RMSE = \sqrt{\frac{\sum_{i=1}^n (\hat{y}_i - y_i)^2}{n}} = 0.1623 \quad (3.7)$$

### 3.3.2 Perancangan Penalaan otomatis PI dengan *Model Reference Adaptive Control* (MRAC)

Kontroler adaptif adalah kontroler yang dapat melakukan penyesuaian terhadap parameter-parameter yang berfungsi mengatur jalannya proses. Kontroler adaptif sering digunakan untuk melakukan penalaan otomatis parameter-parameter kontroler PI. Salah satu yang dapat digunakan adalah metode *Model Reference Adaptive Control* (MRAC).



**Gambar 3.8** Diagram sistem metode penalaan otomatis PI dengan MRAC

Metode MRAC merupakan metode yang berfungsi untuk mendapatkan hasil karakteristik sistem sesuai dengan model referensi yang diinginkan. Algoritma dari metode kontrol ini adalah dengan melakukan perbandingan keluaran sistem dengan model referensi yang dilakukan. Perbandingan tersebut akan menjadi masukan untuk melakukan algoritma adaptif seperti penalaan parameter PI secara

otomatis menyesuaikan kondisi yang ada. Diagram blok dari sistem pengaturan tersebut ditampilkan pada Gambar 3.8.

#### A. Notasi-notasi pada MRAC

Dari hasil pemodelan sistem, diperoleh persamaan matematis sistem seperti pada Persamaan 3.8.

$$\frac{H_p(s)}{Q_{inp}(s)} = \frac{1.75 \cdot e^{-5.15s}}{109.9s + 1} \quad (3.8)$$

Terdapat beberapa variabel yang harus dicari untuk merancang sistem pengaturan dengan metode MRAC, antara lain *error tracking*, *cost function*, dan *update rule*. Kesalahan, *cost function*, dan *update rule* untuk metode MRAC ditampilkan pada Persamaan 3.9, 3.10, dan 3.11.

$$e = y_{plant} - y_{model} \quad (3.9)$$

Persamaan *error tracking* berfungsi untuk menghitung nilai selisih antara respon sistem dengan respon model yang diinginkan.

$$J(\theta) = \frac{1}{2} e^2(\theta) \quad (3.10)$$

Persamaan *cost function* tersebut berfungsi untuk meminimalkan kesalahan sehingga respon sistem akan mendekati nilai yang diinginkan.

$$\frac{d\theta}{dt} = -\gamma \frac{\delta J}{\delta \theta} = -\gamma e \frac{\delta e}{\delta \theta} \quad (3.11)$$

Persamaan *update rule* tersebut berfungsi untuk melakukan update pada sinyal kontrol secara adaptif sesuai dengan kebutuhan sistem.

### B. Algoritma Adaptasi

Elemen lain yang diperlukan pada perancangan metode kontrol MRAC adalah *error tracking*, yang berfungsi untuk mendeteksi nilai kesalahan dari sistem, di mana kesalahan merupakan perbedaan antara keluaran sistem yang terukur dengan sistem yang diinginkan. Persamaan *error tracking* dapat dilihat pada Persamaan 3.12.

$$e = y_{plant} - y_{mdl} \quad (3.12)$$

Penalaan parameter kontroler PI merupakan hal yang penting untuk mendapatkan respon sistem yang diinginkan. Perubahan karakteristik sistem yang dinamis mengakibatkan perlunya penalaan ulang pada parameter kontroler PI. Metode penalaan otomatis merupakan hal yang sering digunakan untuk mengatasi permasalahan tersebut. Salah satu metode yang dapat dilakukan untuk melakukan penalaan otomatis parameter kontroler PI adalah dengan menggunakan metode MRAC. Jika sebuah sistem orde satu mempunyai fungsi alih seperti pada Persamaan 3.13.

$$G(s) = \frac{Y(s)}{U(s)} = \frac{b}{s + a} \quad (3.13)$$

Di mana Y merupakan keluaran sistem, U merupakan sinyal kontrol, b dan a diasumsikan bilangan positif. Jika diberikan kontroler PI, maka sinyal kontrol dari sistem dapat dituliskan pada Persamaan 3.14.

$$U(s) = K_p \cdot (R(s) - Y(s)) + \frac{K_i}{s} \cdot (R(s) - Y(s)) \quad (3.14)$$

Di mana R merupakan masukan pada sistem. Dengan substitusi Persamaan 3.14 ke Persamaan 3.13, diperoleh Persamaan 3.15.

$$\frac{Y(s)}{R(s)} = \frac{b \cdot K_p s + b K_i}{s^2 + (a + b \cdot K_p)s + b \cdot K_i} \quad (3.15)$$

Berdasarkan Persamaan 3.15, diperoleh pendekatan pada kondisi ideal sistem seperti pada Persamaan 3.16.

$$s^2 + (a + b \cdot K_p)s + b \cdot K_i = s^2 + 2\zeta\omega_n s + \omega_n^2 \quad (3.16)$$

Dengan mengacu pada Persamaan 3.16, maka digunakan model referensi orde satu dengan fungsi alih seperti pada Persamaan 3.17.

$$\frac{Ym(s)}{R(s)} = \frac{bm}{am_1s + am_2} = \frac{1}{s + 1} \quad (3.17)$$

Persamaan untuk *error tracking* pada sistem dituliskan pada Persamaan 3.18.

$$e = Y - Ym \quad (3.18)$$

Persamaan 3.18 dapat dituliskan ulang menjadi Persamaan 3.19.

$$e = \frac{b \cdot K_p s + b \cdot K_i}{s^2 + 2\zeta\omega_n s + \omega_n^2} \cdot (r - y) - \frac{bm}{am_1s + am_2} \cdot r \quad (3.19)$$

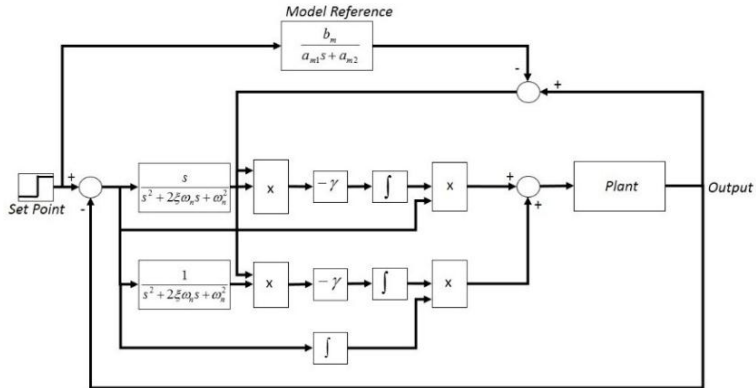
Dengan menggunakan aturan *MIT rule*, diperoleh aturan adaptasi untuk setiap parameter kontroler seperti pada Persamaan 3.20 dan Persamaan 3.21.

$$\frac{dK_p}{dt} = -\gamma \frac{\partial J}{\partial K_p} = -\gamma e \frac{bs}{s^2 + 2\zeta\omega_n s + \omega_n^2} (r - y) \quad (3.20)$$

$$\frac{dK_i}{dt} = -\gamma \frac{\partial J}{\partial K_i} = -\gamma e \frac{\partial e}{\partial K_i} = -\gamma e \frac{b}{s^2 + 2\zeta\omega_n s + \omega_n^2} (r - y) \quad (3.21)$$

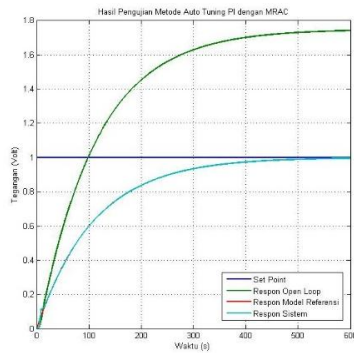
Dari algoritma adaptasi tersebut, sistem pengaturan dengan metode penalaan otomatis parameter kontroler PI menggunakan MRAC dapat

direpresentasikan pada diagram blok. Diagram blok representasi dari sistem pengaturan dengan metode penalaan otomatis parameter kontroler PI menggunakan MRAC ditampilkan pada Gambar 3.9.



**Gambar 3.9** Diagram blok untuk algoritma penalaan otomatis parameter PI

Respon dari model diinginkan dibuat menyerupai orde 1 dengan *gain overall* 1 dan tanpa ada waktu tunda. Dengan adanya algoritma adaptasi penalaan otomatis parameter kontroler PI tersebut, respon dari sistem diharapkan akan mengikuti respon dari model referensi.



**Gambar 3.10** Respon pengujian metode penalaan otomatis PI dengan MRAC

Setelah mendapatkan diagram blok dari algoritma penalaan otomatis dari parameter PI, nilai dari parameter-parameter pada sistem disesuaikan dengan kebutuhan dari kriteria sistem. Hasil dari simulasi dengan memasukkan parameter-parameter sistem ditampilkan pada Gambar 3.10

Dari pengujian menggunakan dua metode penalaan parameter PI, dari Gambar 3.7 dan 3.10 didapatkan hasil bahwa kedua metode tersebut mempunyai kelebihan dan kekurangan. Pada kedua metode didapatkan respon sistem dengan kesalahan waktu tunak 0%. Perbandingan untuk kedua metode ditampilkan pada Tabel 3.5. Berdasarkan perhitungan kesalahan, diperoleh RMSE untuk closed *loop* sistem tersebut seperti pada Persamaan 3.22 dan Persamaan 3.23..

$$RMSE = \sqrt{\frac{\sum_{i=1}^n (\hat{y}_i - y_i)^2}{n}} = 0.0265 \quad (3.22)$$

Persamaan 3.22 merupakan perhitungan RMSE terhadap model.

$$RMSE = \sqrt{\frac{\sum_{i=1}^n (\hat{y}_i - y_i)^2}{n}} = 0.3226 \quad (3.23)$$

Persamaan 3.23 merupakan perhitungan RMSE terhadap masukan sistem.



*Halaman ini sengaja dikosongkan*

## BAB 4

### PENGUJIAN DAN ANALISIS

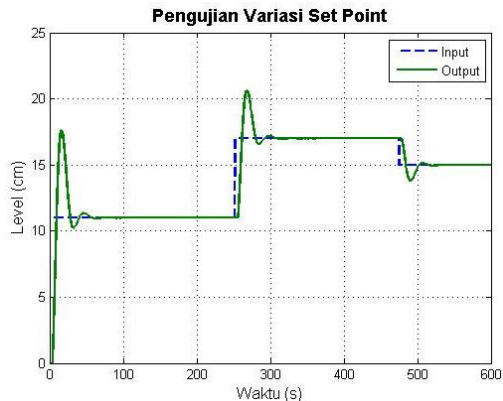
Dengan adanya perbedaan respon yang diperoleh dari kedua metode tersebut perlu dilakukan pengujian dengan berbagai variasi kondisi untuk membandingkan kelebihan dan kekurangan dari dua metode tersebut, dan untuk memilih metode yang terbaik untuk diterapkan pada sistem pengaturan level pada level *supervisory control*.

#### 4.1 Simulasi Pengujian Variasi Keadaan pada Penalaan PI Cohen Coon

Pada metode kontrol ini dilakukan pengujian dalam berbagai kondisi variasi *set point*, variasi parameter kontroler, variasi derau dan *disturbance* pada sistem untuk mengetahui keandalan dari metode kontrol ini.

##### 4.1.1 Pengujian Variasi *Set point*

Pada pengujian ini dilakukan percobaan dengan tiga nilai variasi *set point*. Variasi *set point* yang digunakan adalah *set point* 1, *set point* 7, dan *set point* 5. Hasil simulasi pengujian variasi *set point* pada metode control PI hasil penalaan PI ditampilkan pada Gambar 4.1.

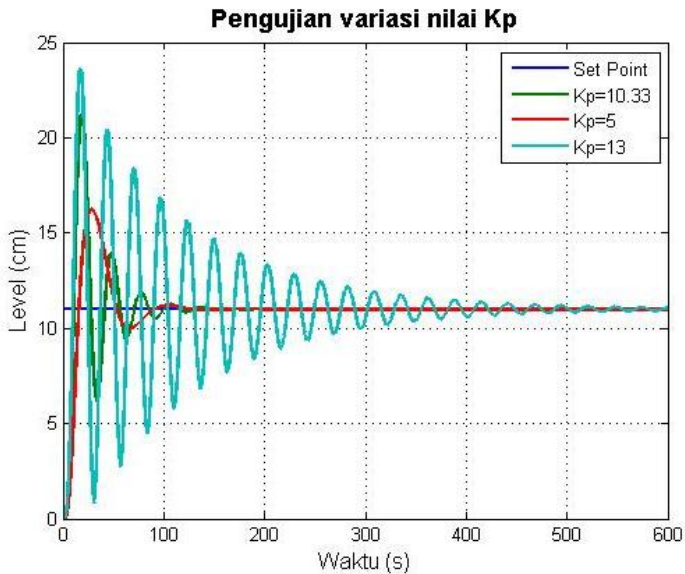


**Gambar 4.1** Hasil pengujian variasi *set point* pada metode control PI hasil penalaan dengan penalaan otomatis

Dari Gambar 4.1 dapat diamati bahwa dengan adanya pengaruh perubahan *set point* pada sistem, dengan adanya algoritma dari kontroler PI, respon dari sistem akan tetap menuju ke nilai dari *set point*. Hal tersebut diakibatkan adanya konstanta proporsional yang mempercepat respon dan konstanta integrator yang akan menghilangkan kesalahan waktu tunak pada respon sistem.

#### 4.1.2 Pengujian Variasi $K_p$

Pengujian berikutnya dilakukan dengan merubah-ubah nilai parameter dari konstanta proporsional untuk mengetahui efek dari konstanta proporsional terhadap keluaran dari sistem. Hasil pengujian dari variasi  $K_p$  ditampilkan pada Gambar 4.2.



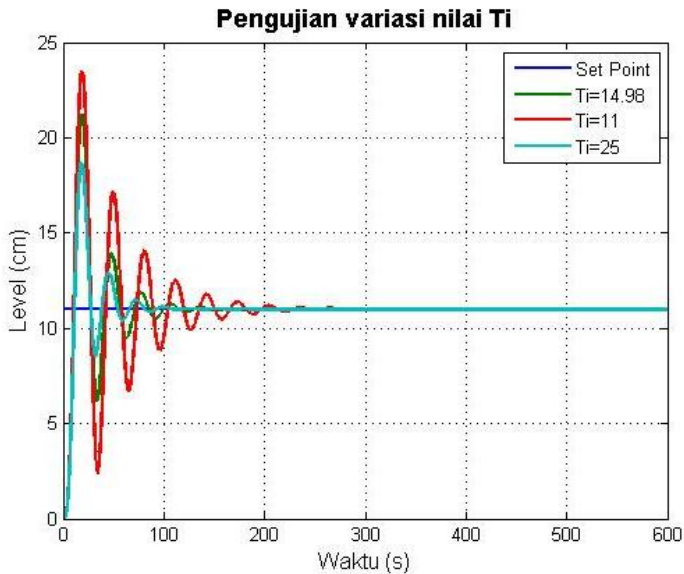
**Gambar 4.2 Hasil pengujian variasi nilai  $K_p$  pada metode PI penalaan Cohen Coon**

Dari hasil pengujian variasi nilai  $K_p$  dapat diamati bahwa sesuai dengan teori, semakin besar nilai konstanta proporsional, respon dari

sistem akan semakin cepat pula. Konstanta proporsional tersebut akan semakin mengurangi kesalahan waktu tunak.

#### 4.1.3 Pengujian Variasi $T_i$

Pengujian berikutnya dilakukan dengan merubah-ubah nilai parameter dari konstanta proporsional untuk mengetahui efek dari konstanta proporsional terhadap keluaran dari sistem. Hasil pengujian dari variasi  $T_i$  ditampilkan pada Gambar 4.3.



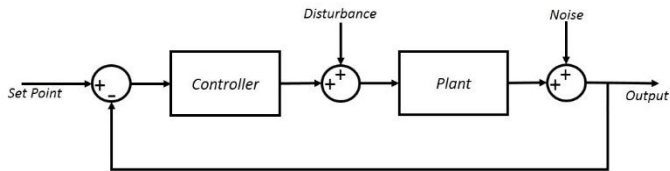
**Gambar 4.3 Pengujian variasi  $T_i$  pada metode PI penalaan Cohen Coon**

Berdasarkan hasil pengujian, sesuai dengan teori, dengan penambahan nilai konstanta integrator, hal tersebut akan menghilangkan nilai kesalahan waktu tunak pada sistem, sehingga respon dari sistem akan menuju ke nilai yang diinginkan.

#### 4.1.4 Pengujian Variasi Derau

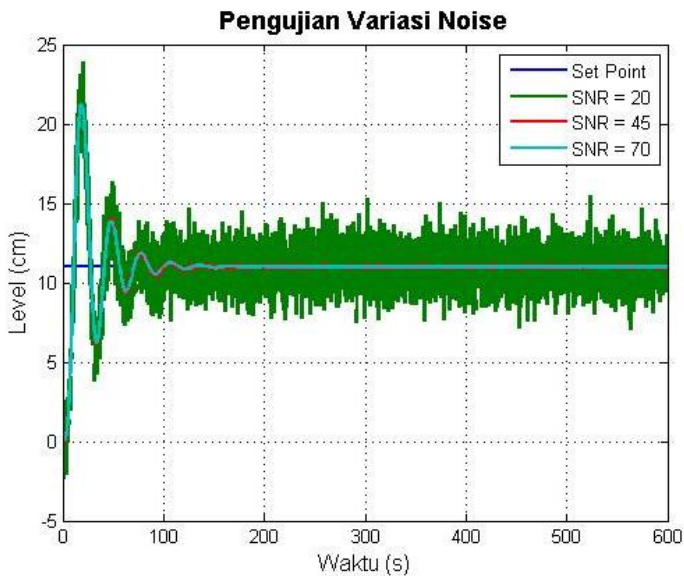
Pengujian dengan melakukan penambahan derau dilakukan untuk mengetahui tingkat kestabilan dari sistem apabila diberi gangguan pada

masukannya maupun keluarannya dari sistem. Diagram blok untuk penambahan derau dan *disturbance* ditampilkan pada Gambar 4.4



**Gambar 4.4** Diagram blok untuk penambahan derau dan *disturbance* pada sistem

Pengujian dengan derau dilakukan dengan menambahkan derau *white* Gaussian. Penambahan derau *white* Gaussian dilakukan dengan penambahan beberapa variasi *Signal to Noise Ratio* (SNR). Hasil pengujian dengan penambahan derau ditampilkan pada Gambar 4.5

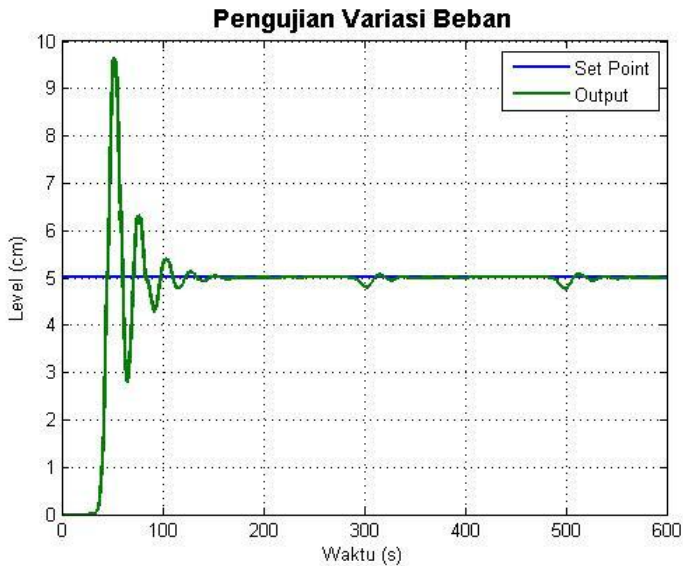


**Gambar 4.5** Hasil pengujian dengan penambahan variasi derau pada sistem

Dari Gambar 4.5 Dapat diamati bahwa semakin besar perbandingan SNR, maka pengaruh dari derau ke sistem akan semakin kecil. SNR merupakan perbandingan antara nilai daya dari sinyal dengan nilai daya dari derau.

#### 4.1.5 Pengujian Variasi Beban

Untuk pengujian dengan menggunakan beban, pengujian dilakukan dengan memberikan sinyal gangguan pada masukan sistem. Dengan pemberian sinyal gangguan tersebut, nilai dari sinyal kontrol yang masuk ke sistem akan berubah dan mengakibatkan perubahan respon pada sistem. Hasil dari pengujian penambahan beban ditampilkan pada Gambar 4.6.



**Gambar 4.6 Hasil penambahan variasi *disturbance* pada sistem**

Berdasarkan Gambar 4.6, dapat diamati bahwa dengan adanya algoritma dari kontroler PI, beban dari sistem dapat dihilangkan. Hal tersebut diakibatkan oleh adanya konstanta proporsional dan konstanta integrator. Permasalahan yang terjadi adalah waktu kembali yang masih

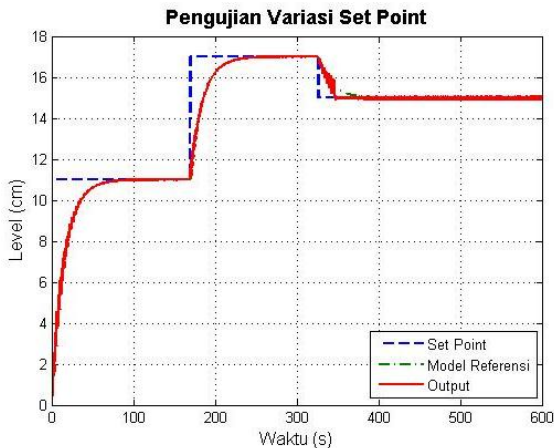
terlalu lama, yaitu sekitar 20 detik. Dengan waktu kembali yang masih lama tersebut, diperlukan metode yang dapat mengurangi pengaruh pembebanan sehingga akan mempercepat waktu kembali.

## 4.2 Simulasi Pengujian Variasi Keadaan pada Metode Penalaan otomatis PI dengan MRAC

Pada metode kontrol ini dilakukan pengujian dalam berbagai kondisi variasi *set point*, variasi parameter kontroler, variasi derau dan *disturbance* pada sistem untuk mengetahui keandalan dari metode kontrol ini.

### 4.2.1 Pengujian Variasi *Set point*

Pada pengujian ini diberikan nilai *set point* yang berbeda-beda ke sistem. Variasi *set point* yang diberikan bernilai sama dengan pengujian pada kontroler PI dengan metode penalaan Cohen Coon. Hasil pengujian dengan variasi *set point* pada metode penalaan otomatis PI menggunakan MRAC ditampilkan pada Gambar 4.7.



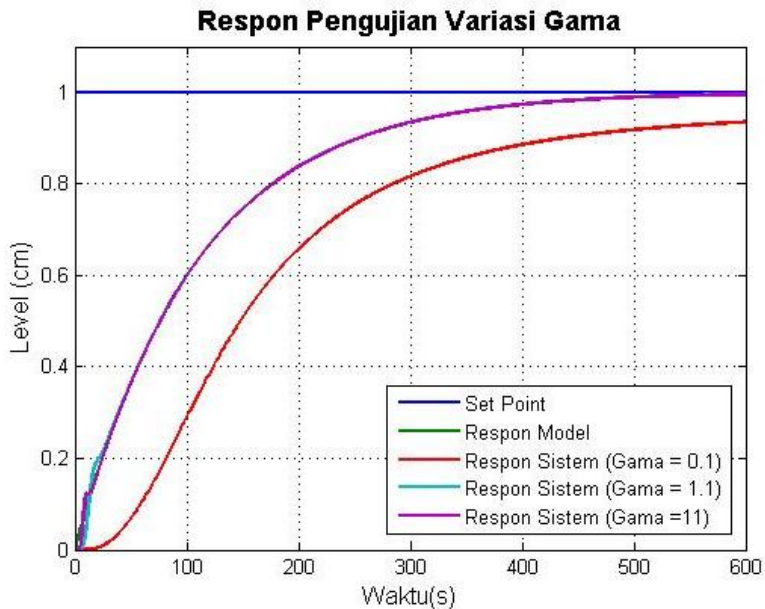
**Gambar 4.7** Hasil pengujian variasi *set point* pada metode penalaan otomatis dengan MRAC

Berdasarkan hasil pengujian, algoritma adaptasi dapat menghasilkan respon sesuai dengan yang diinginkan meskipun nilai *set point* berubah.

Hal tersebut diakibatkan adanya algoritma adaptasi yang menyesuaikan nilai  $K_p$  dan  $K_i$  secara otomatis. Di sisi lain, proses adaptasi pada sistem membutuhkan waktu yang lama, sehingga *rise time* yang dibutuhkan lebih lama jika dibandingkan kotroler PI dengan penalaan Cohen Coon.

#### 4.2.2 Pengujian Variasi Konstanta Adaptasi

Pengujian variasi konstanta adaptasi dilakukan untuk menentukan nilai adaptive gain ( $\gamma$ ) yang sesuai dengan kebutuhan sistem.



**Gambar 4.8 Hasil pengujian variasi konstanta adaptasi**

Pada pengujian ini diberikan nilai konstanta adaptasi yang berbedabeda ke sistem. Pengujian ini bertujuan untuk mengetahui pengaruh konstanta adaptasi terhadap respon dari sistem. Hasil pengujian untuk variasi konstanta adaptasi ditampilkan pada Gambar 4.8.

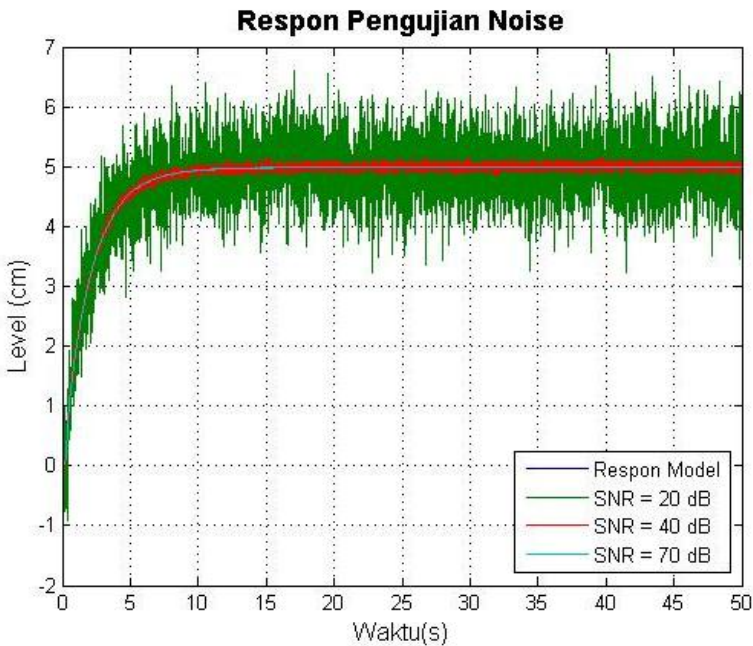
Sesuai dengan teori, semakin besar nilai konstanta adaptasi yang diberikan, maka respon dari sistem akan lebih cepat untuk beradaptasi,



tetapi jika nilai konstanta adaptasi terlalu besar, hal tersebut dapat mengakibatkan respon sistem menjadi tidak stabil.

#### 4.2.3 Pengujian Variasi Derau

Pengujian dengan penambahan derau dan *disturbance* pada sistem dilakukan dengan kondisi sama seperti pengujian derau dan *disturbance* pada metode kontrol PI dengan penalaan Cohen Coon. Hasil pengujian penambahan derau ditampilkan pada Gambar 4.9.

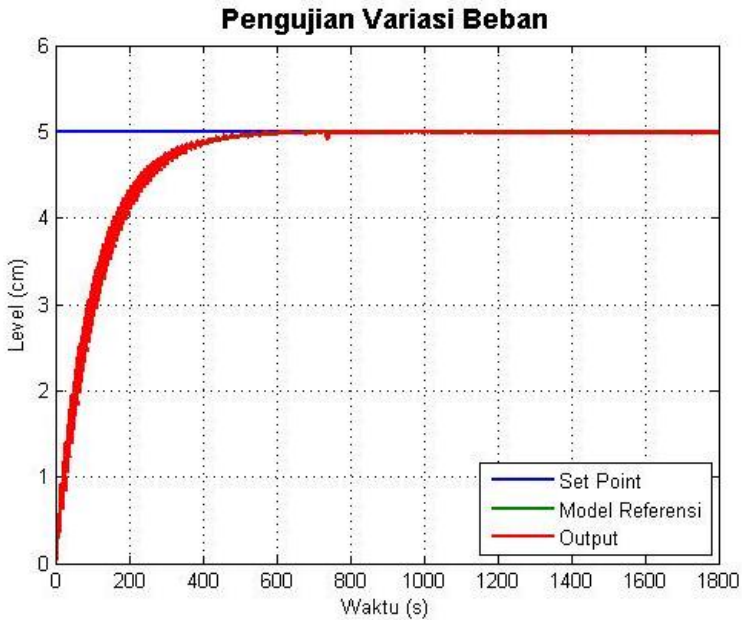


**Gambar 4.9 Hasil pengujian penambahan variasi derau pada sistem dengan metode control penalaan otomatis PI menggunakan MRAC**

Dari Gambar 4.9, dapat dilihat bahwa semakin besar perbandingan SNR, maka pengaruh dari derau ke sistem akan semakin kecil. Berdasarkan perhitungan RMSE, metode penalaan otomatis kontroler PI dengan MRAC memiliki nilai yang lebih besar jika dibandingkan dengan kontroler PI dengan penalaan aturan Cohen Coon.

#### 4.2.4 Pengujian Variasi Beban

Pada pengujian variasi beban untuk metode control penalaan otomatis dengan MRAC, dilakukan pengujian yang sama dengan metode control PI penalaan Cohen Coon. Hasil pengujian variasi beban untuk metode penalaan otomatis dengan MRAC ditampilkan pada Gambar 4.10.



**Gambar 4.10 Hasil pengujian variasi beban pada metode penalaan otomatis PI**

Dari hasil simulasi dengan berbagai keadaan, akan dianalisa mengenai kesalahan dan tunda dari setiap metode kontrol. Analisa dari kesalahan untuk setiap metode kontrol dilakukan untuk membanding respon dari kedua metode kontrol yang mempunyai karakteristik nilai kesalahan lebih kecil.

#### 4.2.5 Perhitungan RMSE pada Metode Kontrol PI dengan Penalaan Cohen Coon

Dari hasil simulasi yang diperoleh pada percobaan pengaturan level menggunakan kontroler PI hasil penalaan dengan metode Cohen Coon, diperoleh RMS Kesalahan seperti pada Persamaan 4.1

$$RMSE = \sqrt{\frac{\sum_{i=1}^n (\hat{y}_i - y_i)^2}{n}} = 0.1623 \quad (4.1)$$

Dengan perhitungan yang sama, diperoleh RMSE untuk setiap variasi kondisi sistem untuk metode kontrol PI penalaan Cohen Coon seperti pada Tabel 4.1.

**Tabel 4.1. Nilai RMSE pada sistem dengan metode kontrol PI penalaan Cohen Coon untuk setiap variasi kondisi**

No.	Kondisi	Nilai RMSE
1	<i>Set point</i> = 1	0.1623
2	<i>Set point</i> = 7	1.1361
3	<i>Set point</i> = 5	0.8115
4	K <sub>p</sub> = 10.33	0.1623
5	K <sub>p</sub> = 5	0.1474
6	K <sub>p</sub> = 13	0.2721
7	T <sub>i</sub> = 14.98	0.1623
8	T <sub>i</sub> = 11	0.2101
9	T <sub>i</sub> = 25	0.1365
10	Derau = 20 dB	0.3676
11	Derau = 45 dB	0.1623
12	Derau = 70 dB	0.1623
13	Beban minimal	0.9977
14	Beban nominal	0.9939
15	Beban maksimal	0.9913

#### 4.2.6 Perhitungan RMSE pada Metode Penalaan otomatis dengan MRAC

Dari hasil simulasi yang diperoleh pada percobaan pengaturan level menggunakan metode penalaan otomatis PI dengan MRAC, diperoleh

RMS Kesalahan terhadap model seperti pada Persamaan 4.2 dan RMS Kesalahan terhadap *set point* seperti pada Persamaan 4.3.

$$RMSE = \sqrt{\frac{\sum_{i=1}^n (\hat{y}_i - y_i)^2}{n}} = 0.0265 \quad (4.2)$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^n (\hat{y}_i - y_i)^2}{n}} = 0.3226 \quad (4.3)$$

Dengan cara yang sama, dilakukan perhitungan nilai RMSE pada sistem dengan metode kontrol penalaan otomatis PI menggunakan MRAC untuk setiap variasi kondisi. Hasil perhitungan RMSE untuk metode penalaan otomatis PI dengan MRAC ditampilkan pada Tabel 4.2.

**Tabel 4.2. Perhitungan nilai RMSE untuk setiap variasi kondisi**

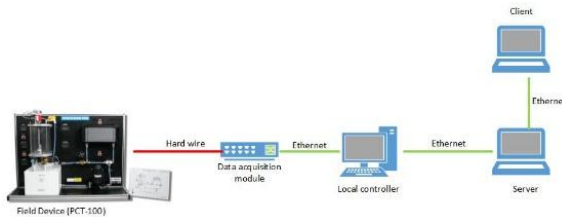
No.	Kondisi	Nilai RMSE Terhadap Model	Nilai RMSE Terhadap Masukan
1	<i>Set point</i> = 1	0.0265	0.3227
2	<i>Set point</i> = 7	0.1113	2.1879
3	<i>Set point</i> = 5	0.0795	1.5628
4	$\gamma = 0.1$	0.1690	0.4460
5	$\gamma = 1.1$	0.0069	0.3027
6	$\gamma = 11$	0.0036	0.3026
7	Derau = 20 dB	0.2698	0.4070
8	Derau = 45 dB	0.0079	0.3048
9	Derau = 70 dB	0.0063	0.3047
10	<i>Beban minimal</i>	0.0156	0.3081
11	<i>Beban nomimal</i>	0.0114	0.3026
12	<i>Beban maksimal</i>	0.0198	0.3023

Berdasarkan data yang diperoleh, nilai kesalahan untuk metode penalaan otomatis dengan MRAC mempunyai nilai RMSE yang lebih

besar dibandingkan dengan PI penalaan Cohen Coon, hal tersebut diakibatkan dari algoritma adaptasi yang membutuhkan waktu lebih lama dibandingkan dengan penalaan Cohen Coon.

### 4.3 Implementasi Kontroler pada Sistem Pengaturan Level

Setelah melakukan berbagai pengujian variasi keadaan untuk kontroler PI hasil rancangan dengan metode penalaan Cohen Coon maupun Penalaan otomatis dengan MRAC, dilakukan implementasi pengaturan level pada plant PCT 100 dengan *set up* pengujian ditampilkan pada Gambar 4.11.

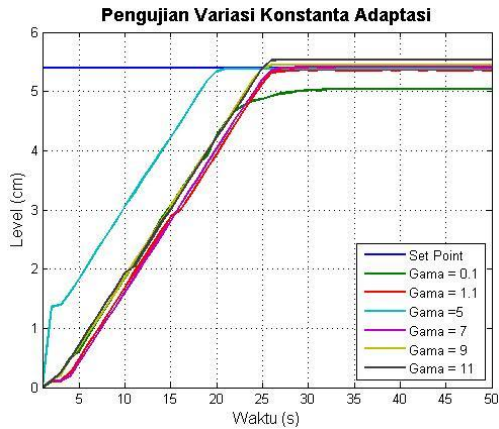


**Gambar 4.11. Set up komponen untuk pengambilan data implementasi sistem pengaturan level**

Pada implementasi yang dilakukan, pengambilan data dilakukan pada level *supervisory control*. Akuisis data dari *field device* dilakukan menggunakan ADAM 5000TCP/6000. Data yang berasal dari *field device* akan dibaca pada *analog input* ADAM, lalu diteruskan ke komputer yang berfungsi sebagai *local controller*. Keluaran dari kontroler akan dikeluarkan oleh *analog output* ADAM dan diteruskan ke aktuator pada *field device*. Pemberian *set point* dan parameter kontroler pada sistem dilakukan melalui level *supervisory control*.

#### 4.3.1 Pengujian Variasi Konstanta Adaptasi

Pengujian variasi konstanta adaptasi dilakukan untuk mendapatkan nilai gama yang paling optimal sehingga didapatkan respon yang paling sesuai. Pada pengujian ini dilakukan beberapa variasi konstanta adaptasi, yaitu 0.1, 1.1, 5, 7, 9, dan 11. Hasil respon untuk pengujian variasi konstanta adaptasi ditampilkan pada Gambar 4.12. Dari hasil pengujian kemudian dipilih nilai konstanta adaptasi dengan respon terbaik.



**Gambar 4.12.** Pengujian variasi konstanta adaptasi pada sistem pengaturan level

Pada hasil pengujian yang dilakukan, didapatkan nilai konstanta adaptasi yang mempunyai *rise time* paling cepat adalah 5, seperti ditunjukkan pada Tabel 4.3.

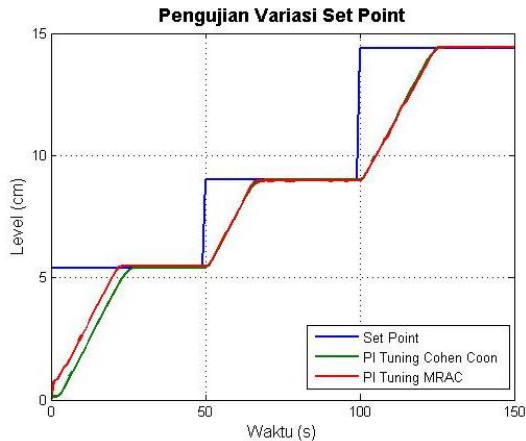
**Tabel 4.3.** Perhitungan *rise time* dan RMSE untuk setiap nilai konstanta adaptasi

No.	Konstanta Adaptasi	<i>Rise time</i>	RMSE
1	0.1	17.8 detik	5.035
2	1.1	19.3 detik	0.042
3	5	12.6 detik	0.034
4	7	19.5 detik	0.046
5	9	19.8 detik	0.053
6	11	19.4 detik	0.063

#### 4.3.2 Pengujian Variasi *Set point*

Pengujian dengan variasi *set point* dilakukan untuk mengamati kemampuan tracking dari metode kontrol PI penalaan Cohen Coon dan metode kontrol penalaan otomatis PI dengan MRAC. Pengujian dilakukan dengan variasi *set point* 5.4, 9, dan 14.4. Hasil respon untuk

kedua metode kontrol dapat mengikuti perubahan *set point* seperti ditampilkan pada Gambar 4.13.



**Gambar 4.13.** Pengujian *set point tracking* pada sistem pengaturan level

Untuk mengetahui efektifitas dari dua metode tersebut, dilakukan perhitungan nilai RMSE dengan hasil ditampilkan pada Tabel 4.4.

**Tabel 4.4.** Perhitungan nilai RMSE untuk variasi *set point*

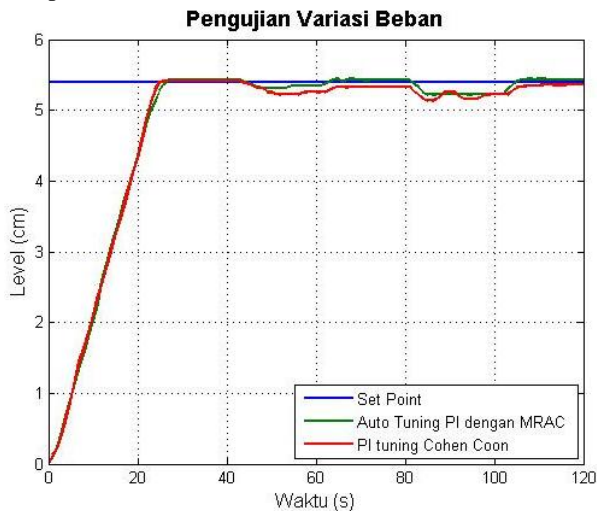
No.	<i>Set point</i>	PI penalaan Cohen Coon	Penalaan otomatis PI
1	5.4	0.0707	0.053
2	9	0.0801	0.058
3	14.4	0.0836	0.067

Pada pengujian variasi *set point*, ketika digunakan kontroler PI dengan tuning Cohen Coon, respon dari pompa sebagai aktuator mengalami *switching* ketika nilai level mendekati nilai *set point*. Hal tersebut diakibatkan karakteristik dari pompa yang menggunakan relay mekanik. Dengan penggunaan relay mekanik tersebut, nilai parameter yang dimasukkan harus sesuai, sehingga metode penalaan otomatis parameter kontroler PI lebih memberikan hasil yang lebih baik.

Pada pengujian yang dilakukan, ketika parameter kontroler yang dimasukkan tidak sesuai, maka pompa akan mengalami switching terus menerus, dan hal tersebut dapat menyebabkan sparking pada kontak relay. Pada dunia industri, hal tersebut sangat membahayakan, dan dapat menyebabkan kebakaran pada sistem. Dengan memperhatikan faktor tersebut, metode tuning parameter kontroler yang digunakan harus dapat memberikan sinyal kontrol yang sesuai.

### 4.3.3 Pengujian Variasi Beban

Pengujian dengan variasi beban dilakukan untuk mengamati kemampuan dari metode kontrol PI penalaan Cohen Coon dan metode kontrol penalaan otomatis PI dengan MRAC untuk menghilangkan pengaruh pembenanan.. Pengujian dilakukan dengan variasi beban minimal,beban nominal, dan beban maksimal. Hasil respon untuk kedua metode kontrol dapat menghilangkan pengaruh perubahan beban seperti ditampilkan pada Gambar 4.14.



**Gambar 4.14. Pengujian *set point* tracking pada sistem pengaturan level**

Untuk mengetahui efektifitas dari dua metode tersebut, dilakukan perhitungan nilai waktu kembali dengan hasil ditampilkan pada Tabel 4.5.



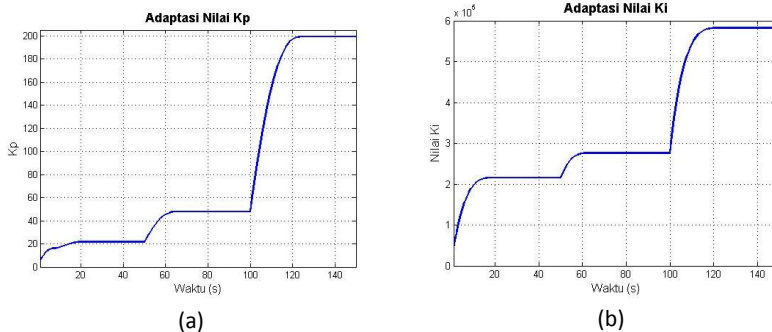
**Tabel 4.5. Perhitungan waktu kembali untuk setiap kondisi beban**

No.	Kondisi beban	PI Penalaan Cohen Coon	Penalaan otomatis PI
1	Nominal	23 detik	18 detik
2	Maksimal	27 detik	23 detik

Pada data yang diperoleh melalui pengujian kedua metode kontrol, sesuai dengan teori, metode kontrol adaptif dapat mengurangi efek pembebanan yang terjadi pada sistem. Hal tersebut diakibatkan oleh adanya algoritma adaptasi yang melakukan penalaan otomatis terhadap parameter kontroler sehingga sinyal kontrol dari sistem akan menyesuaikan kebutuhan sistem pada saat terjadi pembebanan.

#### 4.3.4 Adaptasi Nilai $K_p$ dan $K_i$

Dengan adanya algoritma adaptasi dari metode kontrol penalaan otomatis PI dengan MRAC, nilai dari  $K_p$  dan  $K_i$  akan berubah secara otomatis ketika terjadi perubahan karakteristik sistem seperti ditampilkan pada Gambar 4.15.



**Gambar 4.15. (a) Adaptasi nilai  $K_p$  dan (b) adaptasi nilai  $K_i$  pada metode kontrol penalaan otomatis PI dengan MRAC**

Adanya perubahan otomatis nilai  $K_p$  dan  $K_i$  tersebut akan mengakibatkan sinyal kontrol beradaptasi sesuai dengan kondisi pembebanan pada sistem, sehingga sesuai dengan teori dapat memberikan waktu kembali lebih cepat daripada metode kontroler PI dengan penalaan

aturan Cohen Coon. Pada pengujian yang dilakukan, waktu kembali dari sistem dengan metode kontrol penalaan otomatis PI dengan MRAC mempunyai waktu kembali lebih cepat, yaitu 18 detik pada saat beban nominal, dan 23 detik pada saat beban maksimal. Algoritma adaptasi tersebut juga lebih efektif dalam meminimalkan kesalahan waktu tunak hingga 0.0503.

*Halaman ini sengaja dikosongkan*

## LAMPIRAN

### Lampiran 1. *Script* program simulasi untuk pengujian metode kontroler PID

```
clear all
clc
t=0:0.1:600;

SP=11;
r1=11;
r2=17;
r3=15;

%% Variasi Perubahan Beban
disp('Fungsi alih Waktu Delay')
delay=tf([6],[136.6 79.56 30.9 6])
disp('Fungsi alih Beban 1')
FL1=tf([1.75],[109.9 1])
disp('Fungsi alih Beban 2')
FL2=tf([0.72],[45.216 1])
disp('Fungsi alih Beban 3')
FL3=tf([0.48],[30.144 1])

A=FL1;
C=delay*FL2;
H=delay*FL3;

FCH=SP*step(A,t);

L1=tf([10.33*14.98 10.33],[14.98 0]);
L2=tf([11.48*13.62 11.48],[13.62 0]);
L3=tf([11.20*13.09 11.20],[13.09 0]);

R=feedback(L1*A,1);
I=feedback(L2*C,1);
D=feedback(L3*H,1);

F1=SP*step(R,t);
```

```

F2=SP*step(I,t);
F3=SP*step(D,t);

%% Variasi Set point
disp('Fungsi alih Controller Variasi Set point')
C1=tf([10.33*14.98 10.33],[14.98 0])
C2=tf([10.33*14.98 10.33],[14.98 0])
C3=tf([10.33*14.98 10.33],[14.98 0])

cloop1=feedback(C1*A,1);
cloop2=feedback(C2*A,1);
cloop3=feedback(C3*A,1);

X1=r1*step(cloop1,t);
X1=X1';
Y1=r2*step(cloop2,t);
Y1=Y1';
Z1=r3*step(cloop3,t);
Z1=Z1';

%% Variasi KP
disp('Fungsi alih Controller Variasi Kp')
C11=tf([10.33*14.98 10.33],[14.98 0])
C12=tf([5*14.98 5],[14.98 0])
C13=tf([13*14.98 13],[14.98 0])

cloop11=feedback(C11*A,1);
cloop12=feedback(C12*A,1);
cloop13=feedback(C13*A,1);

X2=SP*step(cloop11,t);
X2=X2';
Y2=SP*step(cloop12,t);
Y2=Y2';
Z2=SP*step(cloop13,t);
Z2=Z2';

%% Variasi Ti

```

```

disp('Fungsi alih Controller Variasi Ti')
C21=tf([10.33*14.98 10.33],[14.98 0])
C22=tf([10.33*11 10.33],[11 0])
C23=tf([10.33*25 10.33],[25 0])

cloop21=feedback(C21*A,1);
cloop22=feedback(C22*A,1);
cloop23=feedback(C23*A,1);

X3=SP*step(cloop21,t);
X3=X3';
Y3=SP*step(cloop22,t);
Y3=Y3';
Z3=SP*step(cloop23,t);
Z3=Z3';

%% Variasi Derau
derau1=awgn(F1,20,'measured');
derau2=awgn(F1,45,'measured');
derau3=awgn(F1,70,'measured');

%% Data dan Plotting
Data(1:6001)=F1;Data(6002:12002)=F2;Data(12003:18001)=F3(1:5999);
Data1(1:6001)=X1; Data1(6002:12002)=Y1;
Data1(12003:18001)=Z1(1:5999);

for k=1:6001
    p(k)=SP;
    p1(k)=r1;
    p2(k)=r2;
    p3(k)=r3;
end
w(1:6001)=p1;w(6002:12002)=p2;w(12003:18001)=p3(1:5999);

for i=1:18001
    IA(i)=SP;

```

```
end
```

```
waktu=0:0.1:1800;
```

```
figure(1)
plot(waktu,Data,waktu,IA,'linewidth',2),grid on;
xlabel('Waktu
(s)','FontWeight','normal','FontSize',12)
ylabel('Level
(cm)','FontWeight','normal','FontSize',12)
title('Respon Variasi
Beban','FontWeight','Bold','FontSize',14)
legend('Keluaran','Set point')
```

```
figure(2)
plot(waktu,w,waktu,Data1,'linewidth',2),grid on;
xlabel('Waktu
(s)','FontWeight','normal','FontSize',12)
ylabel('Level
(cm)','FontWeight','normal','FontSize',12)
title('Pengujian Variasi Nilai Set
point','FontWeight','Bold','FontSize',14)
legend('Set point','Keluaran')
```

```
figure(3)
plot(t,p,t,X2,t,Y2,t,Z2,'linewidth',2),grid on;
xlabel('Waktu
(s)','FontWeight','normal','FontSize',12)
ylabel('Level
(cm)','FontWeight','normal','FontSize',12)
title('Pengujian variasi nilai
Kp','FontWeight','Bold','FontSize',14)
legend('Set point','Kp=10.33','Kp=5','Kp=13')
```

```
figure(4)
plot(t,p,t,X3,t,Y3,t,Z3,'linewidth',2),grid on;
xlabel('Waktu
(s)','FontWeight','normal','FontSize',12)
```

```

ylabel('Level
(cm)', 'FontWeight', 'normal', 'FontSize', 12)
title('Penguajian variasi nilai
Ti', 'FontWeight', 'Bold', 'FontSize', 14)
legend('Set point', 'Ti=14.98', 'Ti=11', 'Ti=25')

figure(5)
plot(t,p,t,deraul,t,derau2,t,derau3, 'linewidth',
2),grid on;
xlabel('Waktu
(s)', 'FontWeight', 'normal', 'FontSize', 12)
ylabel('Level
(cm)', 'FontWeight', 'normal', 'FontSize', 12)
title('Penguajian Variasi
Derau', 'FontWeight', 'Bold', 'FontSize', 14)
legend('Set point', 'SNR = 20', 'SNR = 45', 'SNR =
70')

figure(6)
plot(t,p,t,FCH,t,F1, 'linewidth', 2), grid on;
xlabel('Waktu
(s)', 'FontWeight', 'normal', 'FontSize', 12)
ylabel('Level
(cm)', 'FontWeight', 'normal', 'FontSize', 12)
title('Respon Closed Loop
Sistem', 'FontWeight', 'Bold', 'FontSize', 14)
legend('Set point', 'Respon Tanpa
Kontroler', 'Respon Dengan Kontroler')

figure(7)
plot(t,p,t,FCH, 'linewidth', 2), grid on;
xlabel('Waktu
(s)', 'FontWeight', 'normal', 'FontSize', 12)
ylabel('Level
(cm)', 'FontWeight', 'normal', 'FontSize', 12)
title('Respon Closed Loop
Sistem', 'FontWeight', 'Bold', 'FontSize', 14)
legend('Set point', 'Keluaran')

```



## Lampiran 2. Script program simulasi untuk pengujian metode penalaan otomatis kontroler PI dengan MRAC

```
clear all
clc

disp('Simulasi Pengaturan Level Menggunakan
Metode MRAC')
disp(' ')
disp('Masukkan Parameter')
r=masukan('Set point=');
gamal=masukan('Adaptation Gain 1=')
gama2=masukan('Adaptation Gain 2=')
gama3=masukan('Adaptation Gain 3=')
disp(' ')
disp('Fungsi alih Model Referensi')
%Gm=tf([0.25 7.5947],[1 3.28 7.5947])
Gm=tf([1],[109.9 1])
Ym=r*Gm;
dt=0.01;
m=1;
Iterasi=0:0.01:600;
t=0:0.01:600;
Time=10;
n=60000; %round(Time/dt);
Yp(1:n+1)=1;
Yp1(1:n+1)=1;Yp2(1:n+1)=1;Yp3(1:n+1)=1;
Int11(1:n+1)=0; Int12(1:n+1)=0;
Int21(1:n+1)=0; Int22(1:n+1)=0;
Int23(1:n+1)=0; Int32(1:n+1)=0;
Kesalahan(1:n+1)=0; E(1:n+1)=0;
Kesalahan1(1:n+1)=0; Kesalahan2(1:n+1)=0;
Kesalahan3(1:n+1)=0;
E1(1:n+1)=0; E2(1:n+1)=0; E3(1:n+1)=0;
DKp(1:n+1)=0; DKi(1:n+1)=0;
DKp1(1:n+1)=0; DKp2(1:n+1)=0; DKp3(1:n+1)=0;
DKi1(1:n+1)=0; DKi2(1:n+1)=0; DKi3(1:n+1)=0;
theta(1:1)=1; Kp(1:1)=0; Ki(1:1)=0;
Kp1(1:1)=0; Kp2(1:1)=0; Kp3(1:1)=0;
Ki1(1:1)=0; Ki2(1:1)=0; Ki3(1:1)=0;
```

```

theta1(1:1)=1;theta2(1:1)=1;theta3(1:1)=1;
AA=tf([1],[1 3.28 7.5947]);
sAA=step(AA,t);
sAA=sAA';
AB=tf([1 0],[1 3.28 7.5947]);
sAB=step(AA,t);
sAB=sAB';
Fix=zeros(301,1);
Model=step(Ym,t);
Model=Model';
disp('Fungsi alih Sistem')
Gp=tf([10.5],[15012.34 8880 3475 690.3 6])
sGp=r*theta*Gp;
Yp1=step(sGp,t);
Yp1=Yp1';
Yp2=step(sGp,t);
Yp2=Yp2';
Yp3=step(sGp,t);
Yp3=Yp3';
dis1(1:3000)=0;
dis1(3001:n+1)=1;
dis2(1:3000)=0;
dis2(3001:n+1)=5;
dis3(1:3000)=0;
dis3(3001:n+1)=11;

for i=1:n+1
    p(i)=r;
end

for k=1:1
    for j=1:1
        m=m+1;
        for i=1:60000
            E1(i+1)=p(i)-Yp1(i);
            Kesalahan1(i+1)=Yp1(i)-Model(i);
            DKp1(i+1)=(-
1)*gama1*Kesalahan1(i+1)*sAB(i+1)*E1(i+1);
            Int11(i+1)=(DKp1(i+1)+DKp1(i))*dt/2;

```

```

        Kp1(i+1)=sum(Int11);
        DKi1(i+1)=(-
1) *gama1*Kesalahan(i+1)*sAA(i+1)*E(i+1);
        Int12(i+1)=(DKi(i+1)+DKi(i))*dt/2;
        Int13(i+1)=(E1(i+1)+E1(i))*dt/2;
        IE1(i+1)=sum(Int13);
        Ki1(i+1)=sum(Int12)*IE1(i+1);
        theta1(i+1)=Kp1(i+1)+Ki1(i+1);%-
dis1(i);

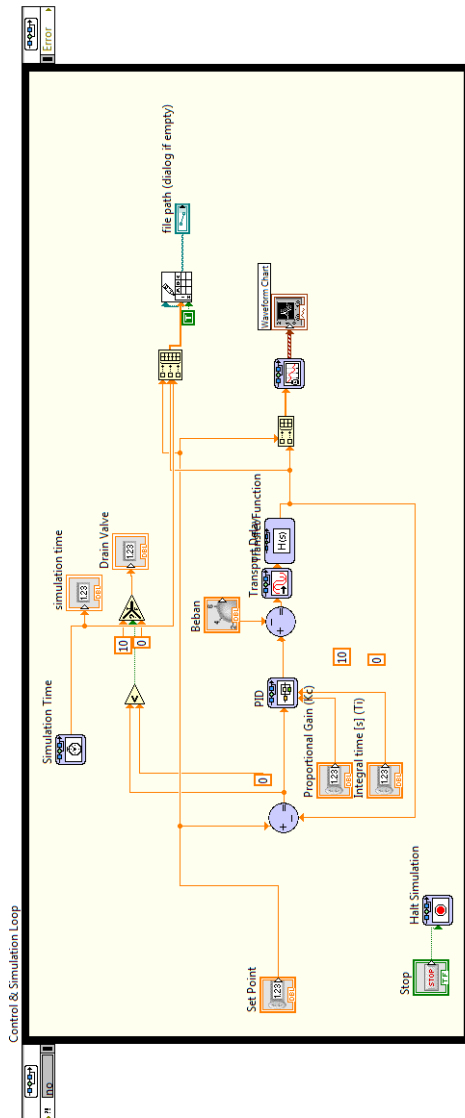
        E2(i+1)=p(i)-Yp2(i);
        Kesalahan2(i+1)=Yp2(i)-Model(i);
        DKp2(i+1)=(-
1) *gama2*Kesalahan2(i+1)*sAB(i+1)*E2(i+1);
        Int21(i+1)=(DKp2(i+1)+DKp2(i))*dt/2;
        Kp2(i+1)=sum(Int21);
        DKi2(i+1)=(-
1) *gama2*Kesalahan2(i+1)*sAA(i+1)*E2(i+1);
        Int22(i+1)=(DKi2(i+1)+DKi2(i))*dt/2;
        Int23(i+1)=(E2(i+1)+E2(i))*dt/2;
        IE2(i+1)=sum(Int23);
        Ki2(i+1)=sum(Int22)*IE2(i+1);
        theta2(i+1)=Kp2(i+1)+Ki2(i+1);%-
dis2(i);

        E3(i+1)=p(i)-Yp3(i);
        Kesalahan3(i+1)=Yp3(i)-Model(i);
        DKp3(i+1)=(-
1) *gama3*Kesalahan3(i+1)*sAB(i+1)*E3(i+1);
        Int31(i+1)=(DKp3(i+1)+DKp3(i))*dt/2;
        Kp3(i+1)=sum(Int31);
        DKi3(i+1)=(-
1) *gama3*Kesalahan3(i+1)*sAA(i+1)*E3(i+1);
        Int32(i+1)=(DKi3(i+1)+DKi3(i))*dt/2;
        Int33(i+1)=(E3(i+1)+E3(i))*dt/2;
        IE3(i+1)=sum(Int33);
        Ki3(i+1)=sum(Int32)*IE3(i+1);
        theta3(i+1)=Kp3(i+1)+Ki3(i+1);%-
dis3(i);

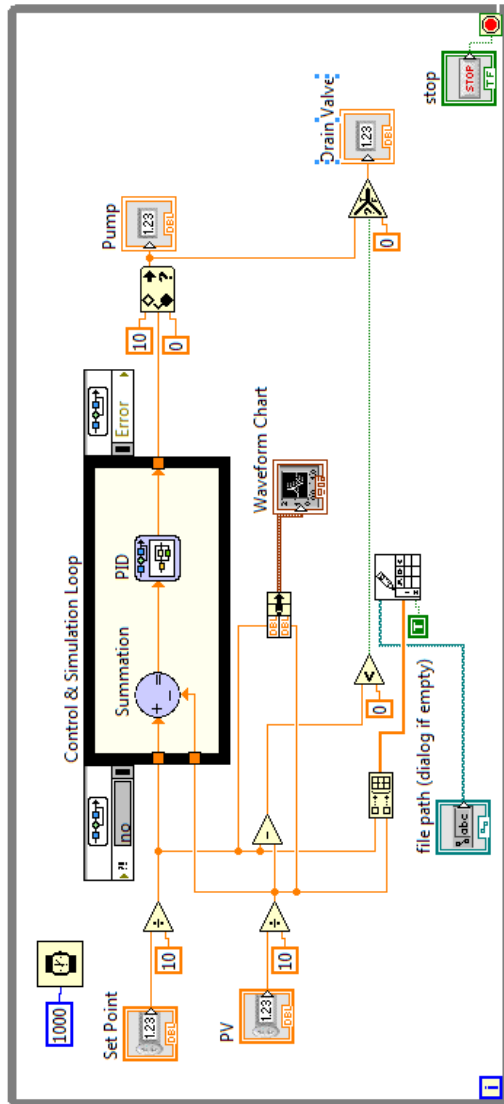
        Yp1(i+1)=theta1(i+1)*Yp1(i+1);
        Yp2(i+1)=theta2(i+1)*Yp2(i+1);
        Yp3(i+1)=theta3(i+1)*Yp3(i+1);

```

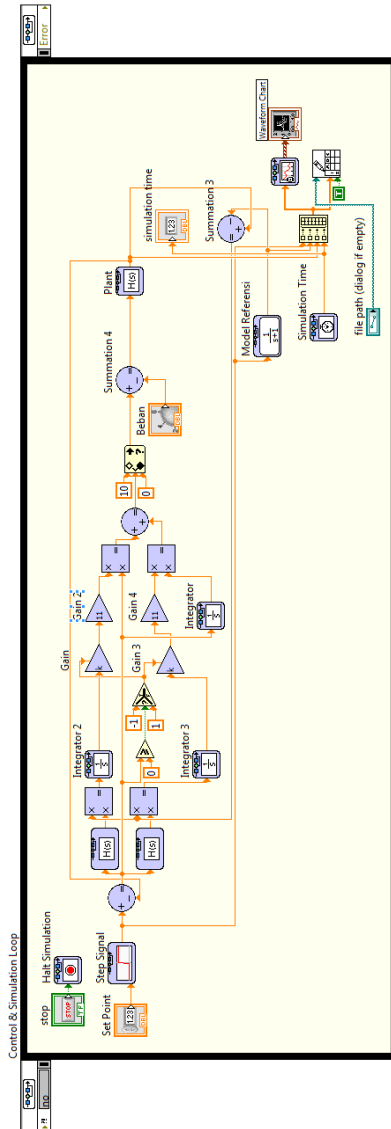
```
        derau1=awgn(Yp1,10,'measured');  
        derau2=awgn(Yp1,50,'measured');  
        derau3=awgn(Yp1,90,'measured');  
    end  
end  
end
```



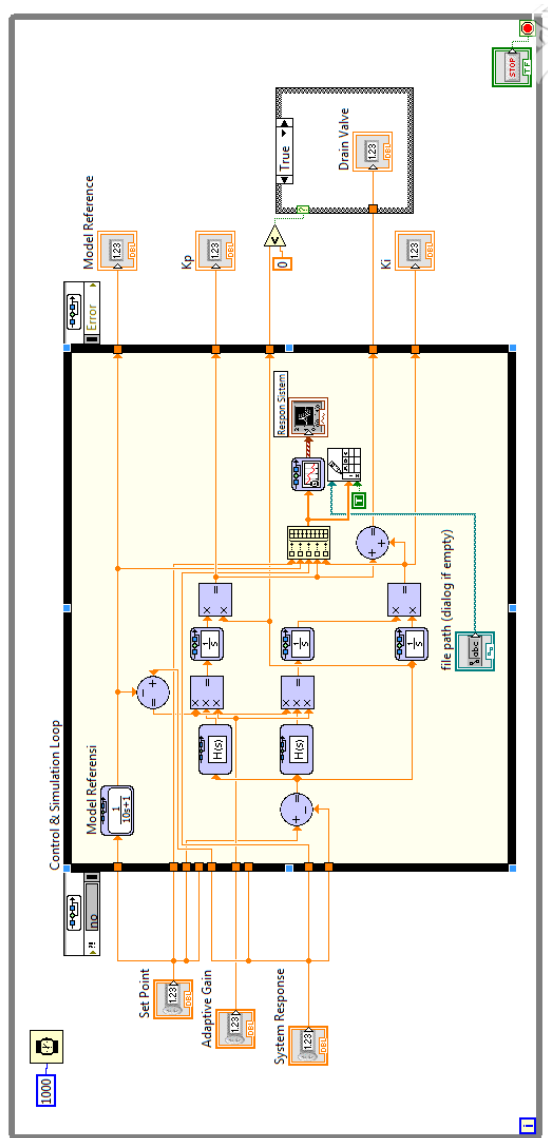
Lampiran 3. Program Simulasi Kontroler PI penalaan Cohen  
Coolh



Lampiran 4. Program Implementasi Kontroler PI penalaan Cohen Coolh



Lampiran 5. Program Simulasi penalaan otomatis PI dengan MRAC



Lampiran 6. Program Implementasi penalaan otomatis PI dengan MRAC



**Lampiran 7. Konfigurasi komunikasi pada Sistem SCADA**

**A. Konfigurasi Koneksi PCT-100 ke ADAM 5000**

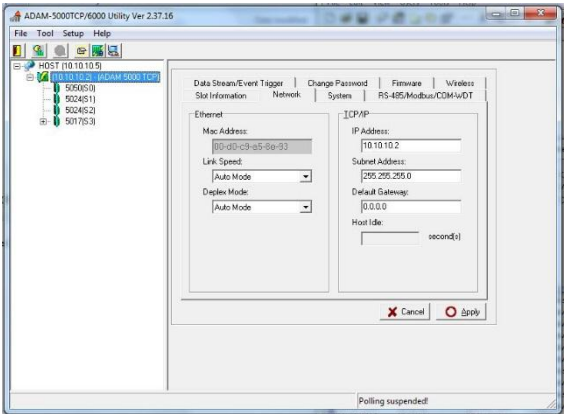
Konfigurasi dari PCT-100 ke modul akuisisi data ADAM 5000 dilakukan menggunakan *software* ADAM-5000 TCP-6000 Utility. Konfigurasi dilakukan dengan menghubungkan komponen-komponen pada *plant* PCT-100 ke *analog* masukan dan *analog* keluaran dari modul ADAM 5000. Konfigurasi dari setiap komponen yang disambungkan ke modul ADAM 5000 ditampilkan pada Tabel A..

**Tabel A.1. Alamat-alamat komponen sistem pengaturan level**

No	Komponen	Alamat	Keterangan
1	Fan	000005	Keluaran digital
2	Diverter Valve	000007	Keluaran digital
3	Pompa	400012	Keluaran analog
4	Drain Valve	400020	Keluaran analog
5	Sensor Level	400032	Masukan analog

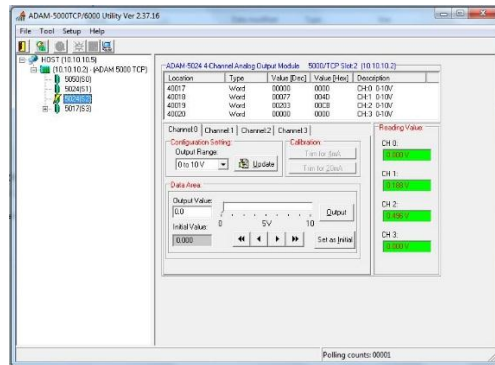
**B. Konfigurasi ADAM ke Local Controller**

Komunikasi dari modul akuisisi data ke PC dilakukan menggunakan TCP/IP. Setting yang dilakukan pada ADAM Utility dilakukan dengan mengatur *IP address* dari modul akuisisi data seperti ditampilkan pada Gambar B.1.



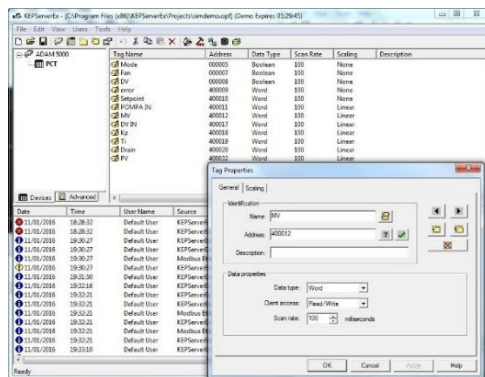
**Gambar B.1. Konfigurasi IP address pada ADAM utility**

Dengan mengacu pada Tabel A., dapat diatur setting konfigurasi dari alamat masukan dan keluaran seperti ditampilkan pada Gambar B.2



**Gambar B.2. Konfigurasi masukan-keluaran sistem pengaturan level**

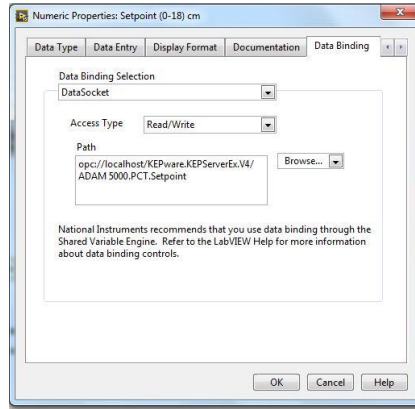
Data dari ADAM Utility dibaca oleh KEPServer yang berperan sebagai OPC Server. Konfigurasi pada KEPServer ditampilkan pada Gambar B.3.



**Gambar B.3. Konfigurasi alamat pada KEPServer**

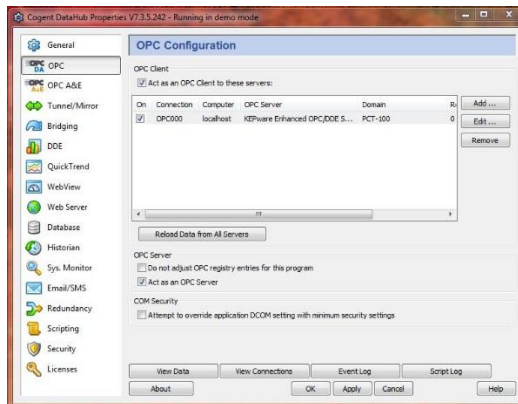
Data dari KEPServer akan dibaca oleh program pengaturan yang dibuat menggunakan LabVIEW pada PC local controller. Pembacaan data dari LabVIEW dilakukan dengan fitur *data binding* pada setiap

komponen masukan dan keluaran dari program seperti ditampilkan pada Gambar B.4



**Gambar B.4. Tools data binding pada pemrograman LabVIEW**

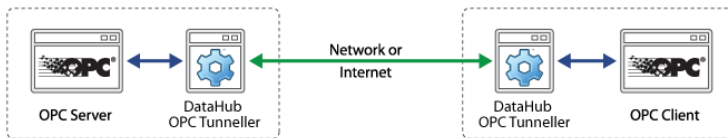
Data pada KEPServer yang berperan sebagai OPC Server akan dibaca pada Cogent DataHub sebagai penghubung ke OPC client. Pembacaan nilai dari OPC Server melalui Cogent DataHub ditampilkan pada Gambar B.5.



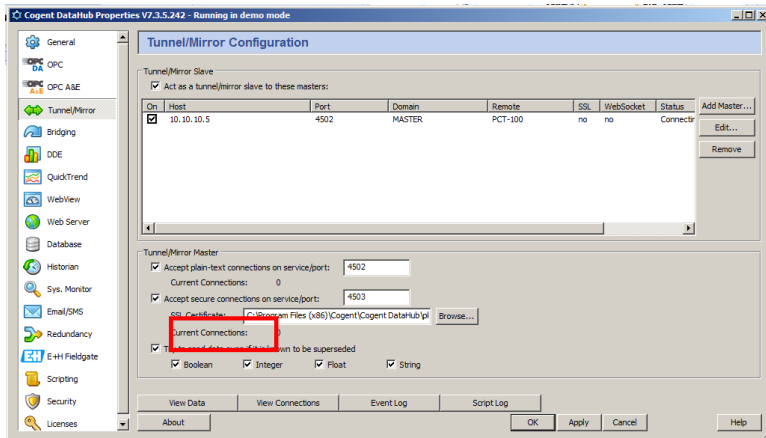
**Gambar B.5. Pembacaan data dari KEPServer pada Cogent DataHub**

### C. Konfigurasi Local Control ke Supervisory Control

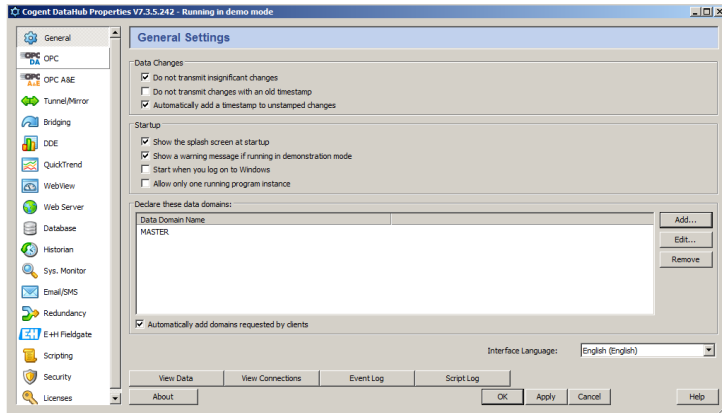
Data dari OPC Server yang dibangun pada PC Based controller dikirim ke komputer server/MTU melalui jaringan. Pengiriman data melalui jaringan dapat dilakukan menggunakan fitur *tunneling*. Salah satu *software* yang dapat digunakan untuk *tunneling* adalah Cogent OPC DataHub. Konsep *tunneling* dapat dilihat pada Gambar C.1. Langkah pertama adalah mengkonfigurasi *master configuration* pada Cogent OPC DataHub dengan cara memilih *Tunnel/Mirror-Add Master*, Gambar C.2, sehingga muncul dialog box “*tunnel/mirror master configuration*”.



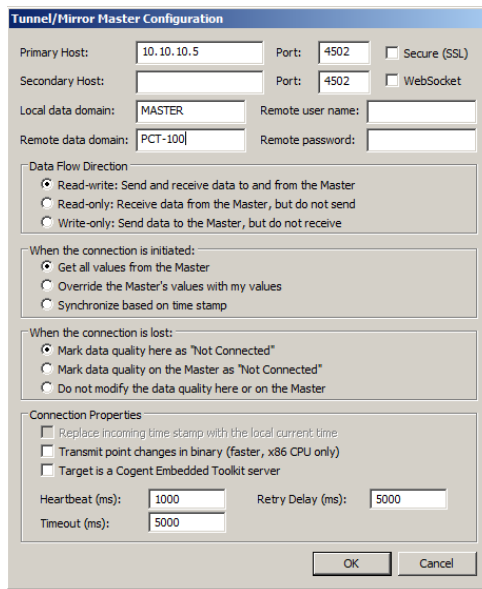
Gambar C.1. Konfigurasi tunneling pada Cogent DataHub



Gambar C.2 Tampilan saat tunneling pada Cogent Datahub



**Gambar C.3. General setting pada Cogent DataHub**

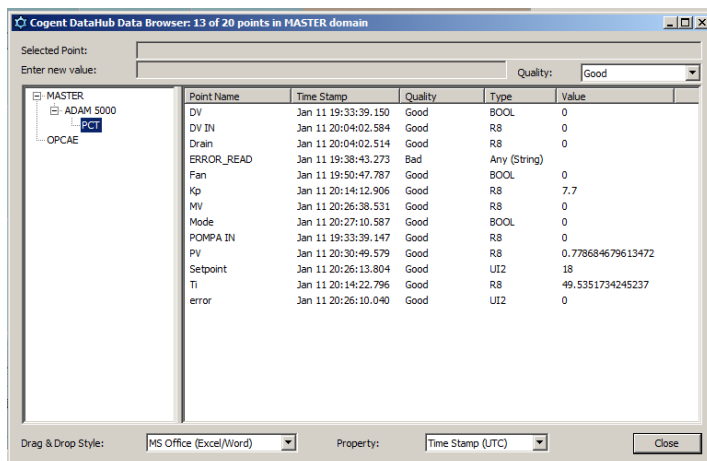


**Gambar C.4. Tampilan untuk mengatur koneksi pada saat melakukan pengujian**

*Primary host* diisi dengan alamat komputer yang berisi OPC Server atau komputer yang berisi data yang ingin diakses, PC Based Controller, Secondary merupakan opsional, jika koneksi ke primary host gagal maka OPC Data Hub secara otomatis akan mengakses *Secondary host*. *Remote data domain* diisi dengan nama domain pada OPC Server, PCT-100. Kotak dialog lain dibiarkan *default*.

Pengecekan koneksi terhadap OPC server dapat dilakukan dengan cara memilih View Data pada *tunnel/mirror*, Gambar C.5, Jika koneksi berhasil maka seluruh *tag* akan memiliki keterangan *good* pada kolom *quality*.

OPC DataHub hanya berfungsi sebagai *tunneler*. Untuk mengakses data pada OPC Server maka diperlukan OPC Client. WinCC merupakan software HMI yang dapat difungsikan sebagai OPC Client Langkah pertama dalam mengakses data pada OPC Server menggunakan WinCC adalah membuka Tag Management pada jendela WinCC Explorer. Jendela WinCC Configuration Studio akan muncul, lalu klik kanan pada OPC Groups, pilih *System Parameter*, sehingga jendela OPC Item Manager akan muncul.



Cogent DataHub Data Browser: 13 of 20 points in MASTER domain

Selected Point:

Enter new value:  Quality:

Point Name	Time Stamp	Quality	Type	Value
DV	Jan 11 19:33:39.150	Good	BOOL	0
DV IN	Jan 11 20:04:02.584	Good	R8	0
Drain	Jan 11 20:04:02.514	Good	R8	0
ERROR_READ	Jan 11 19:38:43.273	Bad	Any (String)	
Fan	Jan 11 19:50:47.787	Good	BOOL	0
Kp	Jan 11 20:14:12.906	Good	R8	7.7
MV	Jan 11 20:26:38.531	Good	R8	0
Mode	Jan 11 20:27:10.587	Good	BOOL	0
POMPA IN	Jan 11 19:33:39.147	Good	R8	0
PIV	Jan 11 20:30:49.579	Good	R8	0.778684679613472
Setpoint	Jan 11 20:26:13.804	Good	UI2	18
Ti	Jan 11 20:14:22.796	Good	R8	49.5351734245237
error	Jan 11 20:26:10.040	Good	UI2	0

Drag & Drop Style:  Property:

**Gambar C.5. Tampilan data view pada Cogent DataHub**

*Halaman ini sengaja dikosongkan*

# BAB 1

## PENDAHULUAN

### 1.1 Latar Belakang

*Supervisory Control and Data Acquisition* (SCADA) merupakan sebuah sistem terdistribusi berbasis komputer yang utamanya digunakan untuk melakukan pengawasan dan pengendalian jarak jauh terhadap kondisi aset di lapangan dari lokasi pusat.[1] Perkembangan dari sistem SCADA sudah sangat pesat, termasuk dalam hal sistem komunikasi yang digunakan. Hingga saat ini, banyak yang menerapkan sistem komunikasi tanpa kabel (*wireless*) atau penggunaan jaringan untuk komunikasi sistem SCADA.

Perkembangan teknologi yang diterapkan pada sistem SCADA tersebut bertujuan untuk meningkatkan efisiensi peralatan yang digunakan dan mempermudah proses akses sistem SCADA tersebut dari berbagai tempat.[2]

Perkembangan teknologi yang pesat tersebut harus diikuti dengan pengetahuan yang mendalam terhadap sistem SCADA bagi para calon insinyur. Permasalahan yang saat ini terjadi adalah minimnya fasilitas pembelajaran mengenai sistem SCADA bagi para calon insinyur. Permasalahan tersebut akan merugikan baik calon insinyur maupun pelaku industri, sehingga diperlukan suatu fasilitas yang dapat digunakan sebagai media pembelajaran sistem SCADA bagi para calon insinyur.

Pada dunia industri, salah satu proses yang banyak diatur pada sistem SCADA adalah sistem pengaturan level pada tangki. Aplikasi pengaturan level pada tangki banyak digunakan pada berbagai jenis industri, mulai dari industri kimia hingga pembangkit listrik. Permasalahan yang terdapat pada sistem pengaturan level adalah adanya kesalahan waktu tunak pada respon *open loop system*. Kesalahan waktu tunak tersebut menyebabkan respon sistem tidak sesuai dengan *set point* yang diberikan ke sistem.

Metode pengaturan untuk mengatasi permasalahan tersebut sudah banyak dikembangkan, antara lain metode kontrol PI. Metode kontrol tersebut banyak digunakan karena kesederhanaan dalam perancangannya dan mampu memberikan respon sesuai dengan kriteria yang diinginkan.[3] Permasalahan dari metode tersebut adalah parameter-parameter kontroler tidak dapat berubah secara otomatis jika terjadi perubahan kondisi sistem. Hingga saat ini, banyak dikembangkan metode kontrol yang dapat melakukan penalaan parameter PI secara otomatis. Kendala lain yang muncul adalah karakteristik sistem yang selalu



berubah-ubah dan tidak dapat diprediksi, sehingga penalaan parameter PI sulit dilakukan.[4]

Dalam Tugas Akhir ini akan dikembangkan metode penalaan parameter PI berbasis metode adaptif untuk mengatur sistem pengaturan level pada sistem SCADA. Metode kontrol tersebut mempunyai algoritma yang dapat menyesuaikan parameter-parameter kontroler PI sesuai dengan kebutuhan sistem sehingga diharapkan dapat memberikan respon tanpa kesalahan waktu tunak dan mampu mengurangi pengaruh pembebanan.

## **1.2 Rumusan Masalah**

Isu-isu mengenai sistem SCADA semakin kompleks seiring berkembangnya teknologi yang digunakan. Permasalahan yang dihadapi oleh mahasiswa sebagai calon insinyur adalah minimnya fasilitas pembelajaran mengenai sistem SCADA. Isu yang berkembang pada sistem SCADA yang paling banyak terjadi adalah mengenai faktor keamanan dan komunikasi data pada saat proses berlangsung.

Pada sistem pengaturan level, terdapat isu yang berkembang seperti adanya kesalahan waktu tunak pada respon sistem loop terbuka. Selain itu, sistem pengaturan level mempunyai karakteristik respon yang dinamis yang diakibatkan perubahan nilai beban.

## **1.3 Batasan Masalah**

Pada Tugas Akhir ini akan dibahas mengenai isu yang terdapat di proses pengaturan level tangki pada sistem SCADA. *Plant* yang akan digunakan untuk merepresentasikan sistem pengaturan level adalah Process Control Technology (PCT) 100. Metode kontrol yang digunakan adalah *Model Reference Adaptive Control* (MRAC). Dengan menerapkan metode tersebut, akan dianalisa mengenai kesalahan respon dan pengaruh pembebanan terhadap sistem.

## **1.4 Tujuan**

Keluaran yang diharapkan dari Tugas Akhir ini adalah sebuah fasilitas media pembelajaran mengenai sistem SCADA bagi para calon insinyur. Pada media pembelajaran tersebut, diharapkan metode kontrol yang diterapkan dapat meminimalkan kesalahan waktu tunak dan mengurangi pengaruh pembebanan pada sistem pengaturan level.

## 1.5 Metodologi

Metodologi yang dilakukan pada Tugas Akhir ini dilakukan dengan urutan tahapan sebagai berikut:

1. Pemodelan sistem pengaturan level  
Pada tahap ini dilakukan penurunan model matematis sistem pengaturan level air pada tangki melalui hukum kekekalan massa.
2. Perancangan sistem SCADA  
Arsitektur dari sistem SCADA yang akan dirancang, detail komponen yang digunakan, serta kebutuhan sistem dilakukan pada tahap ini.
3. Perancangan kontroler  
Tahapan perancangan kontroler dilakukan setelah mengetahui hasil karakteristik respon sistem melalui tahap pemodelan sistem.
4. Implementasi kontroler  
Kontroler yang diperoleh melalui tahap perancangan kontroler diaplikasikan ke sistem SCADA hasil perancangan pada tahap ini.
5. Pengujian dan analisa  
Tahapan pengujian dilakukan setelah tahapan perancangan dan perancangan kontroler. Data yang didapatkan melalui proses pengujian kemudian dianalisa.
6. Penyusunan buku Tugas Akhir  
Tahapan ini dilakukan setelah semua data yang dibutuhkan telah didapatkan dan sudah dilakukan analisa terhadap data-data tersebut.

## 1.6 Sistematika

Pembahasan Tugas Akhir ini dibagi menjadi lima Bab dengan sistematika penulisan sebagai berikut:

### BAB 1 : PENDAHULUAN

Bab ini memuat latar belakang, rumusan masalah, batasan masalah, tujuan, metodologi, sistematika penulisan, dan relevansi.

### BAB 2 : LANDASAN TEORI

Penjelasan tentang tinjauan pustaka, penjelasan umum mengenai *plant* PCT 100, penjelasan mengenai sistem SCADA, kontroler PID, dan kontroler adaptif dimuat dalam Bab ini.

### BAB 3 : PERANCANGAN SISTEM

Desain dan perancangan algoritma kontrol untuk metode kontrol berdasarkan landasan teori pada Bab 2, serta

perancangan arsitektur sistem SCADA untuk pengaturan level dipaparkan pada bab ini.

**BAB 4 : PENGUJIAN DAN ANALISIS**

Pada Bab ini dibahas mengenai data hasil pengujian dan analisa dari data yang diperoleh.

**BAB 5 : PENUTUP**

Kesimpulan yang diperoleh dari penelitian yang dilakukan dan saran untuk penelitian berikutnya dicantumkan pada bagian ini.

## **1.7 Relevansi**

Tugas Akhir ini diharapkan dapat diaplikasikan sebagai media pembelajaran sistem SCADA bagi pada calon insinyur dan dapat diaplikasikan sebagai metode control untuk menghilangkan kesalahan waktu tunak dan pengaruh pembebanan pada sistem pengaturan level.

## BAB 2

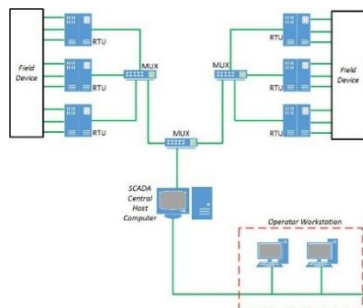
### PENGATURAN LEVEL TANGKI PADA SISTEM SCADA

#### 2.1 Sistem *Supervisory Control and Data Acquisition* (SCADA)

*Supervisory Control and Data Acquisition* (SCADA) merupakan sebuah sistem terdistribusi berbasis komputer yang utamanya digunakan untuk melakukan pengawasan dan pengendalian jarak jauh kondisi pada aset di lapangan dari lokasi pusat. Pada sistem SCADA, proses pengawasan dan pengendalian dilakukan secara terpusat, sehingga akan lebih efisien dan memudahkan operator untuk melakukan proses pengendalian dan pengawasan.[1] Sistem SCADA dirancang dengan tujuan untuk:

1. Pengawasan sistem.
2. Memperoleh kontrol terhadap sistem dan memastikan performansi yang dibutuhkan terpenuhi.
3. Mereduksi jumlah staff operator yang dibutuhkan melalui otomasi atau pengoperasian sistem dari lokasi pusat.
4. Menyimpan data katakteristik sistem.
5. Menampilkan informasi dari performansi sistem dan menentukan prosedur manajemen yang paling efektif.

Secara umum, SCADA tersusun dari komponen-komponen seperti *Field Data Interface Device/Remote Terminal Unit* (RTU), *Master Terminal Unit* (MTU), *Human Machine Interface* (HMI), dan sistem komunikasi. Struktur dari sistem SCADA ditampilkan pada Gambar 2.1.



**Gambar 2.1** Struktur arsitektur dari sistem SCADA

A. *Remote Terminal Unit* (RTU)

RTU berfungsi untuk meng-*interface* peralatan-peralatan yang terdapat di lapangan dan mengkonversikan sinyal dari *field device* ke protokol komunikasi SCADA.

B. *Master Terminal Unit* (MTU)

MTU atau sering juga disebut dengan komputer *server* adalah komputer yang berfungsi memproses informasi antara RTU dan operator.

C. *Human machine interface* (HMI)

HMI merupakan tampilan untuk interaksi antara manusia/operator dengan komputer. Secara umum HMI terletak pada komputer *server*.

D. Sistem Komunikasi

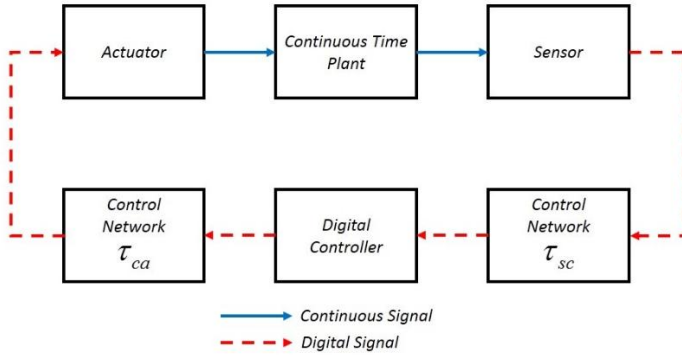
Bagian ini berfungsi sebagai media proses transfer data antara *field interface device*/RTU, *control unit*, dan komputer *server*/MTU. Selain itu, sistem komunikasi digunakan oleh operator, yang lokasinya terpisah secara geografis, untuk mengakses *plant*. Dalam pemilihan sistem komunikasi, perlu diperhatikan berbagai aspek, antara lain:

1. Lokasi dari *field device*
2. Ketahanan dari media komunikasi
3. Ketersediaan pilihan komunikasi
4. Harga dari pilihan komunikasi
5. Ketersediaan sumber daya (*power*)

## 2.2 Sistem Pengaturan Melalui Jaringan

Pada sistem pengaturan berjaringan, informasi berupa *set point*, keluaran, dan sinyal kontrol dikirim menggunakan jaringan dengan komponen-komponen sistem kontrol (sensor, kontroler, aktuator, dll.).[2] Banyak penelitian yang telah dikembangkan menyangkut sistem pengaturan melalui jaringan. Isu yang paling banyak dibahas mengenai sistem pengaturan berjaringan adalah adanya waktu tunda dan *packet drop out*. [2] Pada kenyataannya, sistem pengaturan berjaringan menggunakan sinyal digital untuk dikirimkan ke *plant* yang bersifat kontinyu seperti pada Gambar 2.2.

Pada proses pengiriman data, terjadi dua waktu tunda, yaitu waktu tunda dari sensor menuju kontroler ( $\tau_{sc}$ ), dan waktu tunda dari kontroler menuju aktuator ( $\tau_{ca}$ ).



**Gambar 2.2 Skema komunikasi data pada sistem pengaturan berjaringan**

Dengan adanya dua jenis waktu tunda tersebut, tunda keseluruhan pada sistem pengaturan berjaringan dapat ditulis pada Persamaan 2.1.

$$\tau = \tau_{sc} + \tau_{ca} \quad (2.1)$$

Adanya waktu tunda pada sistem dapat mengakibatkan sinyal kontrol yang berasal dari kontroler akan terlambat masuk ke sistem. Keterlambatan tersebut akan mengakibatkan sinyal kontrol yang diterima sistem menjadi tidak sesuai dengan kebutuhan, sehingga respon dari sistem menjadi tidak sesuai dengan kriteria yang diinginkan.

### 2.3 Pengenalan *Plant* PCT 100 [5]

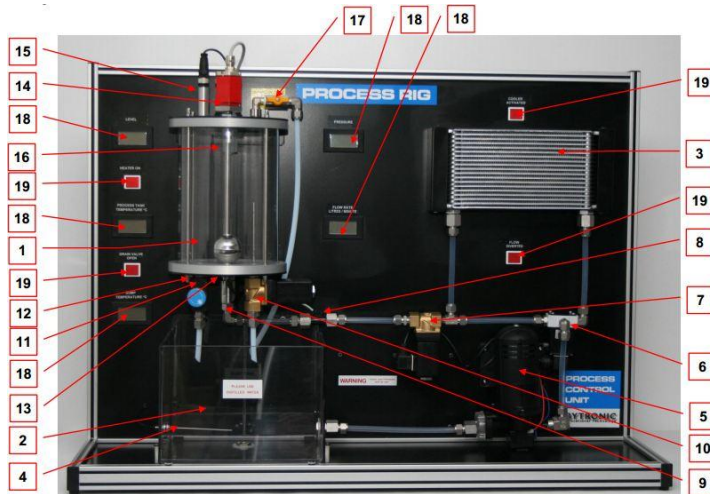
Perancangan sistem dari Tugas Akhir ini dilakukan dengan membuat arsitektur sistem SCADA untuk pengaturan level air tangki. *Plant* yang digunakan pada Tugas Akhir ini adalah *Process Control Technology* (PCT) 100.[5]

PCT 100 merupakan sebuah model sistem yang mengimplementasikan proses kontinyu pada fluida yang merepresentasikan proses yang terjadi di industri. Pada PCT 100, terdapat komponen-komponen, baik berupa *hardware* maupun *software* yang berfungsi untuk melakukan pengaturan pada proses yang terjadi di fluida, di antaranya pengaturan level, *flow*, temperatur, dan tekanan.pada fluida.

Secara umum, PCT 100 terdiri dari dua komponen, yaitu process rig dan control module. Secara umum Process Rig merupakan tempat berlangsungnya proses, sedangkan control module berfungsi untuk melakukan pengaturan pada sistem.

### 2.3.1 Process rig pada PCT 100

Process rig pada PCT 100 berfungsi sebagai tempat berlangsungnya proses pengaturan fluida. Terdapat beberapa komponen pada process rig mulai dari pompa untuk mensirkulasikan air, hingga *heater* dan *fan* untuk mengatur temperatur air. Tampilan dari process rig ditampilkan pada Gambar 2.3.



**Gambar 2.3 Tampilan fisik process rig pada PCT 100[5]**

Process rig pada PCT 100 terdiri dari komponen-komponen yang berfungsi sebagai tempat berlangsungnya pengaturan level, *flow*, temperatur, dan tekanan. Pada Gambar 2.3, terdapat Sembilan belas komponen yang mempunyai fungsi masing-masing. Komponen utama yang terdapat pada Process Rig, seperti pompa dan valve sebagai aktuatur, dan beberapa jenis sensor. Bagian-bagian dari process rig yang terdapat pada Gambar 2.3 ditampilkan pada Tabel 2.1.[5]

**Tabel 2.1 Penjelasan bagian-bagian pada process rig PCT 100[5]**

No	Keterangan
1	<i>Process tank</i>
2	<i>Sump tank</i>
3	<i>Cooler unit</i>
4	Sensor temperatur pada <i>Sump tank (PRT)</i>
5	Pompa
6	<i>3/2 Diverter valve</i>
7	<i>2/2 Proportional drain valve</i>
8	Sensor <i>flow</i>
9	<i>One way check valve</i>
10	<i>2/2 Proportional drain valve</i>
11	<i>Needle valve</i>
12	<i>Pressure relief valve</i>
13	Pemanas
14	Sensor <i>Level</i>
15	<i>Pressure transducer</i>
16	<i>Float switch</i>
17	<i>Overflow/Vent valve</i>
18	<i>Digital LCD displays</i>
19	<i>Indicator lights</i>

### 2.3.2 Control module pada PCT 100

Control Module pada PCT 100 berisi rangkaian–rangkaian elektronik untuk menghubungkan process rig dengan kontroler. Control module pada PCT 100 dilengkapi dengan tombol *fault*, di mana penjelasan untuk setiap fungsi dari setiap tombol *fault* ditampilkan pada Tabel 2.2.

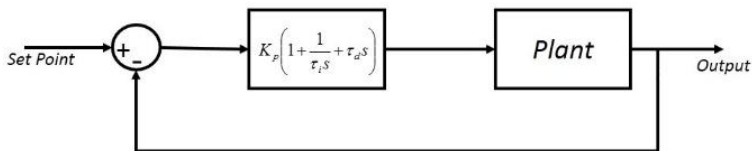
**Tabel 2.2 Penjelasan fungsi *fault switch* pada control module PCT 100[5]**

<i>Fault No.</i>	Fungsi	Efek
F1	<i>A/D Fault</i>	ADC lock up
F2	<i>Pump</i>	Pompa tidak dapat aktif
F3	<i>Temperature</i>	Display pembacaan temperatur maksimal
F4	<i>Cooler</i>	Cooler menyala terus menerus
F5	<i>Heater</i>	Display <i>heat</i> menampilkan pembacaan salah
F6	<i>Flow meter</i>	Tidak ada <i>flow feedback</i>



## 2.4 Kontroler PID

Kontroler PID merupakan kontroler yang paling banyak digunakan pada dunia industri. Karena kebanyakan kontroler PID biasa terdapat pada lokasi *plant*, beberapa metode dikembangkan untuk melakukan penalaan parameter PID secara otomatis. Keunggulan dari kontroler PID terletak pada aplikasi yang luas dan hampir dapat digunakan untuk semua sistem pengaturan.[3] Meskipun terkadang belum memberikan hasil yang optimal, kontroler PID hampir selalu dapat memberikan respon yang baik. Diagram blok dari kontroler PID dapat dilihat pada Gambar 2.4.



**Gambar 2.4 Diagram blok sistem dengan penambahan kontroler PID**

Jenis kontroler PID yang paling banyak diterapkan pada dunia industri adalah kontroler PI, di mana kontroler tersebut merupakan perpaduan dari kontroler proporsional dan kontroler integral. Dengan adanya elemen konstanta proporsional dan konstanta integrator pada sistem, respon sistem, dapat diatur karakteristik *rise time* dan kesalahan waktu tunak dari keluaran sistem.

## 2.5 Metode Penalaan Parameter PID

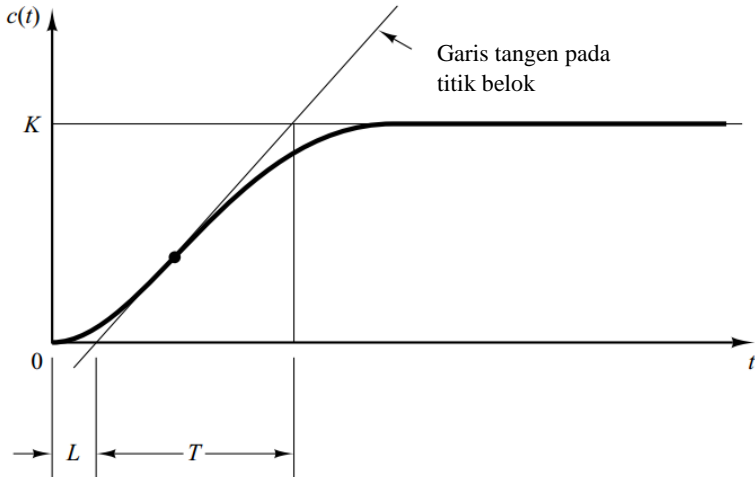
Tujuan utama dari proses penalaan parameter kontroler PID adalah untuk mendapatkan respon yang dianggap paling baik. Kriteria dari sistem yang diharapkan pada umumnya adalah sistem dengan respon cepat dan stabilitas yang tinggi.[1] Terdapat beberapa metode yang dapat digunakan. Dua metode yang paling sering digunakan adalah metode Ziegler Nichols dan Cohen Coon.

### 2.5.1 Aturan Penalaan Ziegler-Nichols

Aturan penalaan ini diperkenalkan oleh Ziegler dan Nichols untuk menentukan nilai *proportional gain*  $K_p$ , *integral time*  $T_i$ , dan *derivative time*  $T_d$  berdasarkan karakteristik respon transien dari suatu *plant*.[3] Terdapat dua metode yang terdapat pada aturan penalaan Ziegler Nichols.

### A. Aturan Penalaan Ziegler Nichols Metode Pertama

Pada metode pertama, respon dari sistem diperoleh dari hasil eksperimen dengan pengujian sinyal *unit step*. Metode ini dapat digunakan jika respon dari sistem menyerupai kurva S seperti pada Gambar 2.5.[3]



**Gambar 2.5** Grafik respon kurva S pada hasil pengujian sistem

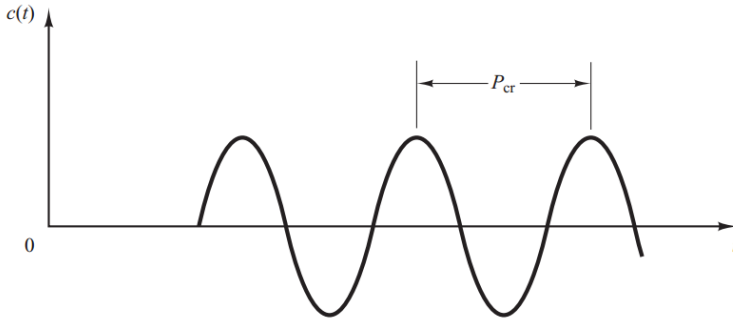
Pada kurva S tersebut terdapat dua karakteristik sistem, yaitu waktu tunda  $L$ , dan *time constant*  $T$ . Penentuan parameter kontroler untuk aturan penalaan Ziegler Nichols metode pertama ditampilkan pada Tabel 2.3

**Tabel 2.3** Tabel aturan penalaan metode pertama Ziegler Nichols

	$K_p$	$T_i$	$T_d$
P	$\frac{T}{L}$	$\infty$	0
PI	$0.9 \frac{T}{L}$	$\frac{L}{0.3}$	0
PID	$1.2 \frac{T}{L}$	$2L$	$0.5L$

### B. Aturan Penalaan Ziegler Nichols Metode Kedua

Pada metode kedua penalaan Ziegler Nichols, nilai awal untuk  $T_i = \infty$  dan  $T_d = 0$ . Dengan hanya menggunakan aksi proporsional, didapatkan nilai kritis  $K_{cr}$  di mana terjadi osilasi pertama dan mempunyai periode  $P_{cr}$  seperti ditunjukkan pada Gambar 2.6.[3]



**Gambar 2.6** Osilasi yang terjadi ketika nilai gain kritis

Perhitungan nilai parameter kontroler PID untuk aturan penalaan untuk metode kedua Ziegler Nichols dilakukan berdasarkan perhitungan sesuai dengan aturan pada Tabel 2.4.

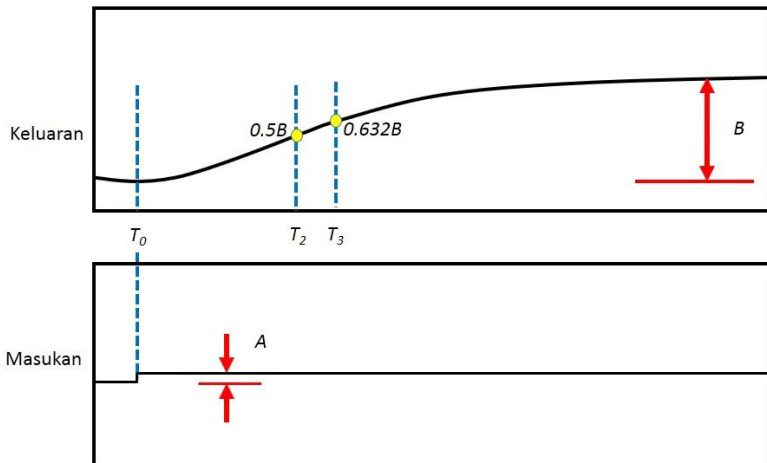
**Tabel 2.4** Tabel aturan penalaan metode kedua Ziegler Nichols

	$K_p$	$T_i$	$T_d$
P	$0.5K_{cr}$	$\infty$	0
PI	$0.45K_{cr}$	$\frac{1}{1.2}P_{cr}$	0
PID	$0.6K_{cr}$	$0.5P_{cr}$	$0.125P_{cr}$

#### 2.5.2 Aturan Penalaan Cohen Coon

Metode penalaan Cohen Coon merupakan metode lain yang banyak digunakan untuk melakukan penalaan parameter PID selain metode Ziegler Nichols. Aturan Cohen Coon mempunyai keunggulan pada lebih

banyaknya variasi proses jika dibandingkan dengan metode Ziegler Nichols. Metode Cohen Coon dapat digunakan pada sistem dengan waktu tunda kurang dari dua kali *time constant*, sedangkan metode Ziegler Nichols hanya dapat digunakan untuk sistem dengan *dead time* kurang dari setengah *time constant*. Prinsip dari aturan penalaan Cohen Coon adalah dengan pendekatan grafik yang menyerupai kurva S seperti pada Gambar 2.7.[2]



**Gambar 2.7. Pengujian step untuk mendapatkan kurva S sebagai dasar penalaan Cohen Coon**

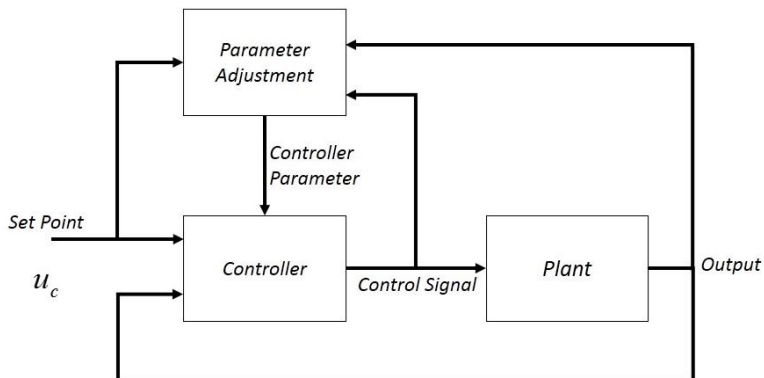
Prinsip dari penalaan parameter kontroler PID dengan metode Cohen Coon tidak jauh berbeda dengan aturan penalaan PID dengan menggunakan aturan Ziegler Nichlos metode pertama. Perbedaan keduanya terletak pada penentuan nilai parameter kontroler dan parameter lain seperti  $T_0$ ,  $T_2$  dan  $T_3$ . Parameter  $T_0$  merupakan waktu ketika masukan diberikan ke sistem,  $T_2$  merupakan waktu ketika respon sistem mencapai 50 % dari waktu tunak, dan  $T_3$  merupakan waktu ketika respon sistem mencapai 63,2 % dari waktu tunak. Aturan untuk penalaan parameter kontroler PID menggunakan metode Cohen Coon ditampilkan pada Tabel 2.5.

**Tabel 2.5 Tabel aturan penalaan metode Cohen Coon**

	$K_p$	$\tau_i$	$\tau_d$
<b>P</b>	$\frac{1}{rK} \left( 1 + \frac{r}{3} \right)$		
<b>PI</b>	$\frac{1}{rK} \left( 0.9 + \frac{r}{12} \right)$	$\tau_{del} \frac{30 + 3r}{9 + 20r}$	
<b>PID</b>	$\frac{1}{rK} \left( \frac{4}{3} + \frac{r}{4} \right)$	$\tau_{del} \frac{32 + 6r}{13 + 8r}$	$\tau_{del} \frac{4}{11 + 2r}$

## 2.6 Kontroler Adaptif

Secara intuisi, kontroler adaptif dapat diartikan sebagai kontroler yang dapat melakukan modifikasi karakteristik dan respon yang berubah-ubah sesuai dengan perubahan proses dan gangguan yang ada. Sistem adaptif sendiri dapat didefinisikan sebagai segala sistem fisik yang dirancang dengan sudut pandang adaptif. Kontroler adaptif juga dapat diartikan sebagai kontroler dengan parameter-parameter yang dapat disesuaikan dan sebuah mekanisme untuk menyesuaikan parameter-parameter tersebut.[2]

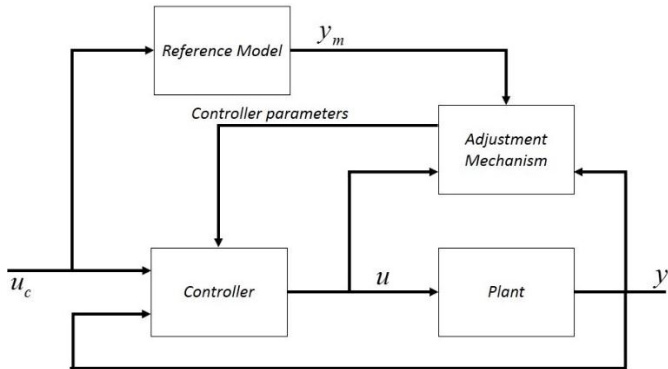


**Gambar 2.8 Diagram blok metode sistem pengaturan adaptif**

Dengan adanya mekanisme tersebut, kontroler akan bersifat *non-linear*, oleh karena itu penggunaan kontroler adaptif dapat digolongkan ke dalam kategori sistem *non-linear*. Sebuah sistem kontrol adaptif dapat mempunyai dua *loop*. *Loop* pertama berfungsi sebagai *feedback* dari sistem dan masuk ke kontroler, *loop* yang lain berfungsi sebagai *loop* untuk menyesuaikan parameter-parameter pada sistem. *Loop* untuk menyesuaikan parameter mempunyai karakteristik lebih lambat dibandingkan *loop feedback* biasa. Diagram blok dari sistem pengaturan adaptif dapat dilihat pada Gambar 2.8.[2]

## 2.7 Model Reference Adaptive System (MRAS)

*Model Reference Adaptive System* (MRAS) merupakan salah satu metode kontrol adaptif yang penting. Sistem ini dapat diasumsikan sebagai sistem adaptif di mana performansi dari sistem yang diinginkan diekspresikan dalam bentuk sebuah model referensi, yang memberikan respon yang diinginkan ke sinyal kontrol.[2] Blok diagram dari MRAS ditampilkan pada Gambar 2.9.[2]



**Gambar 2.9 Diagram blok dari MRAS**

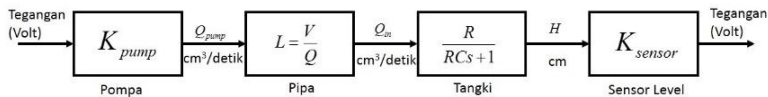
Pada sistem ini terdapat dua *feedback* sebagai sebagai *feedback* dari proses untuk masukan kontroler (*inner loop*) dan sebagai *feedback* untuk adaptasi parameter kontroler (*outer loop*). Mekanisme untuk mengatur parameter pada MRAS dapat dilakukan dengan dua cara, yaitu dengan menggunakan metode pendekatan gradien atau dengan menerapkan teori stabilitas.

## 2.8 Pemodelan Matematis Sistem Pengaturan Level

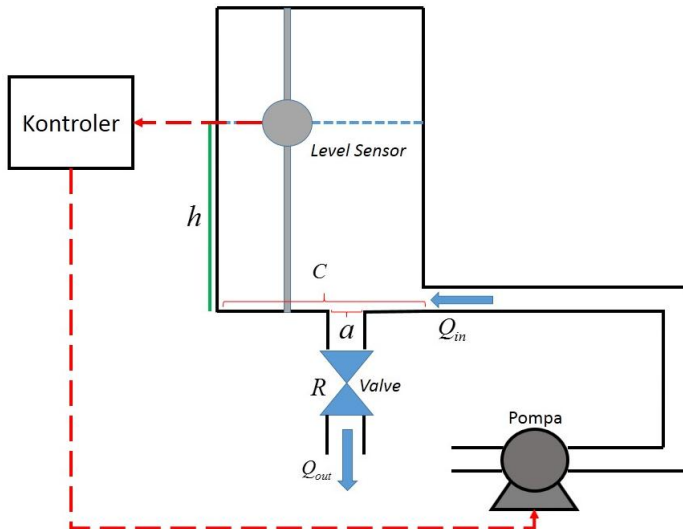
Perancangan sistem selanjutnya dilakukan dengan melakukan pemodelan matematis dari sistem pengaturan level cairan pada tangki. Secara umum prinsip dari pengaturan level pada tangki adalah dengan menjaga debit air yang masuk ke tangki bernilai sama dengan debit air yang keluar dari tangki.

Sistem pada pengaturan level air tangki pada dasarnya berupa sistem orde satu. Tangki diibaratkan seperti kapasitor di mana air mengalir dari dalam dan ke dalam tangki.

Proses yang terjadi pada pengaturan level air tangki dapat dilihat seperti ilustrasi pada Gambar 2.10.



Gambar 2.10. Skema proses pengaturan level pada tangki



Gambar 2.11. Ilustrasi sistem pengaturan level pada tangki

Berdasarkan hasil pemodelan sistem, diperoleh parameter-parameter sistem seperti pada Tabel 2.6.

**Tabel 2.6 Parameter-parameter pada sistem**

No.	Parameter	Nilai
1	Diameter	16 cm
2	C	200.96 cm <sup>2</sup>
3	Q <sub>0</sub>	13.7 cm <sup>3</sup> /detik
4	h <sub>0</sub>	7.5 cm
5	R	0.547 detik/cm <sup>2</sup>
6	a	0.785 <sup>2</sup>

### 2.8.1 Pemodelan Gain Pompa

Pada sistem terdapat pompa dengan *gain* seperti pada Persamaan 2.2.

$$K_{pump} = \frac{3.5 \cdot 1000 / 60}{10} = 5.83 \text{ cm}^3/\text{detik.volt} \quad (2.2)$$

### 2.8.2 Pemodelan Sistem Pengaturan Level Pada Tangki

Prinsip kerja dari pengaturan level adalah menjaga agar aliran air yang masuk dan keluar dari tangki besarnya sama sesuai hukum kekekalan massa, sehingga sesuai dengan Persamaan 2.3.

$$Q_{in} = Q + Q_{out} \quad (2.3)$$

Persamaan 2.3 dapat ditulis ulang menjadi Persamaan 2.4.

$$Q_{in} - Q_{out} = C \frac{dh}{dt} \quad (2.4)$$

Di mana C merupakan kapasitansi tangki yang merupakan perubahan volume air tiap perubahan ketinggian air. Jika luas penampang konstan, maka nilai dari kapasitansi juga konstan, yaitu nilai luas alas dari tangki. Pada sistem terdapat *valve* dengan nilai hambatan R yang merupakan perubahan ketinggian tiap perubahan debit. Nilai dari R dapat dihitung pada Persamaan 2.5.



$$R = \frac{h_0}{Q_0} \quad (2.5)$$

Di mana  $h_0$  merupakan ketinggian level pada titik kerja, dan  $Q_0$  adalah debit air pada titik kerja. Dengan memasukkan nilai R, Persamaan 2.5 dapat ditulis ulang menjadi Persamaan 2.6.

$$Q_{in} - \frac{h}{R} = C \frac{dh}{dt} \rightarrow RQ_{in} - h = RC \frac{dh}{dt} \quad (2.6)$$

Dengan menggunakan transformasi Laplace maka diperoleh Persamaan 2.7.

$$RQ_{in}(s) - H(s) = RCsH(s) \quad (2.7)$$

Dari Persamaan 2.7 diperoleh fungsi alih pada tangki seperti pada Persamaan 2.8.

$$\frac{H(s)}{Q_{in}(s)} = \frac{R}{RCs + 1} \quad (2.8)$$

Dari Tabel 2.6, didapatkan persamaan fungsi alih dari *open loop* sistem seperti pada Persamaan 2.9.

$$\frac{H(s)}{Q_{in}(s)} = \frac{0.547}{109.9s + 1} \quad (2.9)$$

### 2.8.3 Pemodelan Gain pada Sensor

Pada sistem terdapat sensor yang mempunyai penguatan seperti pada Persamaan 2.10.

$$K_{sensor} = \frac{10}{18} = 0.55 \text{ volt/cm} \quad (2.10)$$

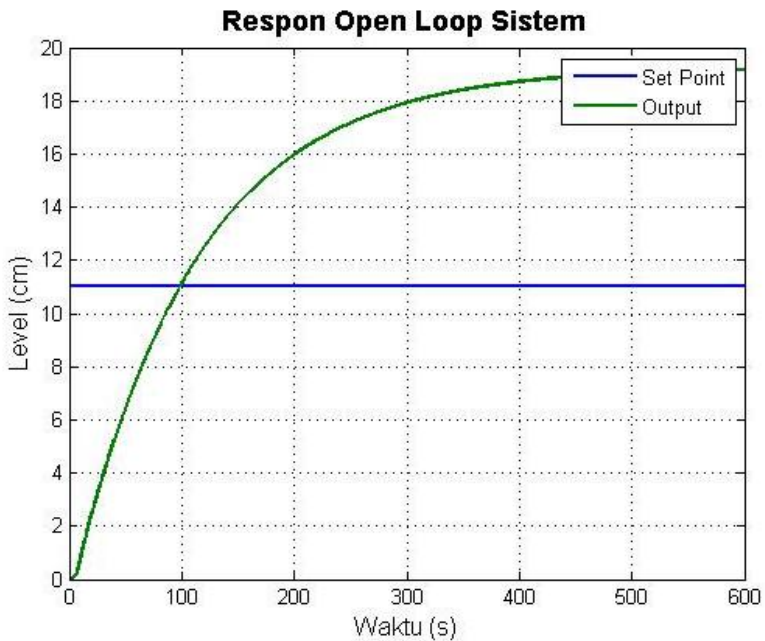
Dengan menggabungkan seluruh elemen pada sistem, fungsi alih keseluruhan sistem dapat dihitung melalui Persamaan 2.11.

$$G(s) = K_{pump} \cdot e^{-Ls} \cdot \frac{R}{RCs + 1} \cdot K_{sensor} \quad (2.11)$$

Dengan memasukkan setiap parameter, diperoleh fungsi alih dari keseluruhan sistem seperti pada Persamaan 2.12.

$$G(s) = 5.83 \cdot e^{-5.15s} \cdot \frac{0.547}{109.9s + 1} \cdot 0.55 = \frac{1.75 \cdot e^{-5.15s}}{109.9s + 1} \quad (2.12)$$

Respon *open loop* sistem ditampilkan pada Gambar 2.12.



**Gambar 2.12** Respon sistem pengaturan level *loop* terbuka

Berdasarkan respon *loop* terbuka yang diperoleh dari pemodelan matematis sistem, masih terdapat kesalahan waktu tunak yang besar, yaitu 0.75. Dengan adanya permasalahan tersebut, perlu digunakan kontroler untuk meminimalkan kesalahan waktu tunak pada sistem.

Pada sistem tersebut juga diperlukan kontroler yang dapat mengurangi efek pembebanan yang berasal dari perubahan nilai beban  $R$ . Perubahan nilai tersebut disebabkan oleh perubahan bukaan *control valve* pada sistem.

## **BAB 5**

### **PENUTUP**

#### **5.1 Kesimpulan**

Metode kontrol penalaan otomatis PI dengan MRAC lebih efektif dalam menghilangkan kesalahan dibandingkan dengan kontroler PI biasa dengan RMSE 5.03%

Metode penalaan otomatis PI dengan MRAC lebih efektif dalam menghilangkan pengaruh pembebanan dibandingkan dengan kontroler PI biasa, yaitu 18 detik pada saat beban nominal, dan 23 detik pada saat beban maksimal.

Algoritma adaptasi pada sistem SCADA dapat mempermudah operator dalam melakukan pengendalian dan pengawasan proses karena dapat melakukan penalaan parameter kontroler PID secara otomatis.

#### **5.2 Saran**

Pada peneleitian selanjutnya dapat dilakukan dengan menggunakan konfigurasi *plant* yang lebih bervariasi pada arsitektur sistem. Selain itu, dapat dilakukan pemilihan nilai konstanta adaptasi menggunakan metode yang lebih baik dibanding penentuan nilai konstanta adaptasi secara eksperimental.

## DAFTAR PUSTAKA

- [1] Bailey, David; Edwin Wright, "*Practical SCADA for Industry*", Linacre House, Oxford, 2003.
- [2] \_\_\_\_\_, "*The Fundamental of SCADA*", Bentley Systems, Incorporated, 2004.
- [3] Ogata, Katsuhiko, "*Modern Control Insinyuring*", Prentice Hall, New Jersey, 1970.
- [4] Astrom, K.J.; Bjorn Wittenmark, "*Adaptive Control 2<sup>nd</sup> Edition*", Dover Publication, Inc, New York, 1995.
- [5] Educational Technology, "PCT-100 Manual v1.62", Bytronic Technology.
- [6] Lin, Feng; Robert D. Brant, and George Saikalis, "*Self-Penalaan of PID Controllers by Adaptive Interaction*", Proc. American Control Conference, pp. 3676-3681, Chicago, Illinois, USA, 2000.
- [7] Ibrahim, Dogan, "*Microcontroller Based Applied Digital Control*", John Wiley & Sons, Ltd, West Sussex, England, 2006.
- [8] Adrian, Coman; Axente Corneliu; Boscoianu Mircea, "*The Simulation of the Adaptive Systems Using the MIT Rule*", Proc. International Conference on Mathematical Methods and Computational Techniques in Electrical Insinyuring. Sofia, Bulgaria. 2008.
- [9] Pirabakaran, K.; V. M. Becerra, "*Automatic Penalaan PID Controllers Using Model Reference Adaptive Control Techniques*", The 27<sup>th</sup> Annual Conference of the IEEE Industrial Electronics Society, Denver, CO, 2001.
- [10] Singh, Baljinder; Vijay Kumar, "*Design and Simulation of Penalaan otomatis of PID Controller using MRAC Technique for Coupled Tanks System*", International Journal of Science and Research, 2013.

## BIODATA PENULIS



Adetya Devritama lahir di Bantul, 1 Agustus 1993. Penulis telah menyelesaikan pendidikan formal di SD Muhammadiyah Bodon, SMP Negeri 5 Yogyakarta, dan SMA Negeri 1 Yogyakarta. Pada tahun 2012, penulis melanjutkan studi ke Jurusan Teknik Elektro, Fakultas Teknologi Industri, Institut Teknologi Sepuluh Nopember Surabaya dan mengambil fokus Bidang Studi Teknik Sistem Pengaturan. Selama di bangku kuliah, penulis tergabung sebagai anggota Unit Kegiatan Mahasiswa (UKM) Robotika ITS dan aktif sebagai anggota *Automation and Industrial Robotics Research Group*. Pada tahun 2016, penulis mengikuti seminar dan ujian Tugas Akhir sebagai salah satu persyaratan untuk memperoleh gelar Sarjana Teknik.